PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-067867
(43)Date of publication of application : 11.03.1994

(51)Int.Cl. G06F 9/06
G06F 12/00

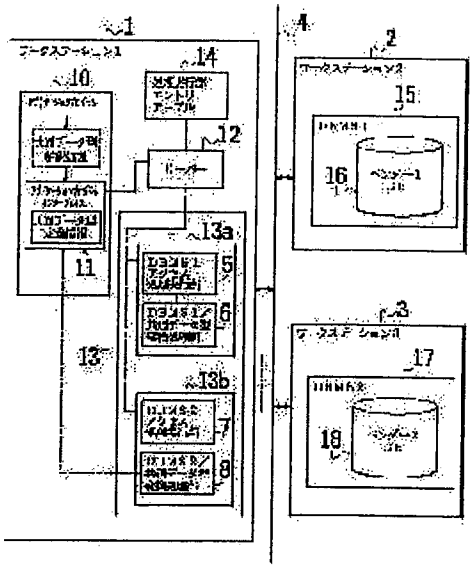
(21)Application number : 04-350043
(22)Date of filing : 03.12.1992

(71)Applicant : FUJI XEROX CO LTD
(72)Inventor : HIGANO MICHIO

(30)Priority
Priority number : 04178859 Priority date : 15.06.1992 Priority country : JP

(54) DATA BASE ACCESSING SYSTEM FOR APPLICATION PROGRAM

(57)Abstract:
PURPOSE: To provide an application program data base access system capable of accessing from the same application program to plural data base management systems(DBMSs) by a common access module.
CONSTITUTION: The data base access system is provided with an application program interface 11 for executing interface processing independent of the sort of a DBMS, respective processing execution parts 13 for executing access processing to respective DBMSs, and a server 12 for connecting the processing execution part 13 for executing access processing to an objective DBMS to the application program based upon a connection request from the interface 11.



(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平6－67867

(43)公開日　平成６年(1994)３月11日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　9/06	４３０　Ｅ	9367－5B		
12/00	５１３　Ｊ	8526－5B		

審査請求　未請求　請求項の数３(全 18 頁)

(21)出願番号　特願平4－350043

(22)出願日　平成４年(1992)12月３日

(31)優先権主張番号　特願平4－178859

(32)優先日　平４(1992)６月15日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005496
富士ゼロックス株式会社
東京都港区赤坂三丁目３番５号

(72)発明者　日向野　道夫
神奈川県川崎市高津区坂戸３丁目２番１号　ＫＳＰ　Ｒ&Ｄビジネスパークビル　富士ゼロックス株式会社内

(74)代理人　弁理士　南野　貞男　(外３名)

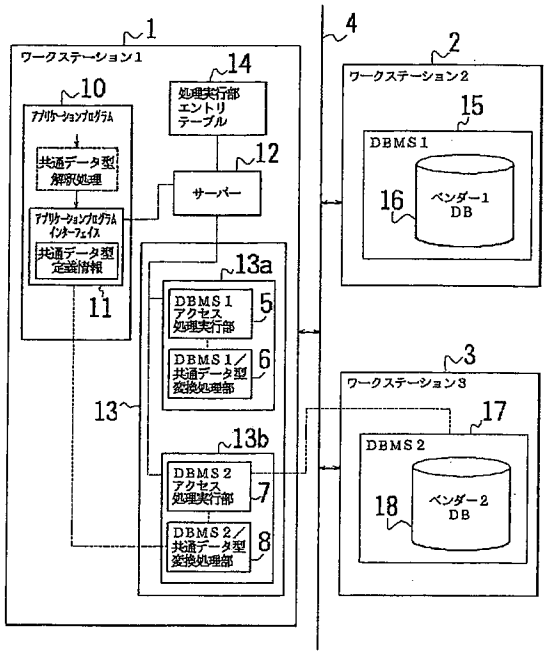
(54)【発明の名称】　アプリケーションプログラムのデータベースアクセス方式

(57)【要約】

【目的】複数のＤＢＭＳへ同一のアプリケーションプログラムからのアクセスが共通のアクセスモジュールで可能となるアプリケーションプログラムのデータベースアクセス方式を提供する。

【構成】データベース管理システムの種別に依存しないインタフェース処理を行うアプリケーションプログラムインタフェースと、複数の各々のデータベース管理システムに対してそれぞれにアクセス処理を行う各処理実行部と、前記アプリケーションプログラムインターフェイスからの接続要求に基づいて、対象のデータベース管理システムに対するアクセス処理を行う処理実行部とアプリケーションプログラムと間の接続を行うサーバーとを備える。

図1



【特許請求の範囲】

【請求項１】　アプリケーションプログラムにおいてデータベース管理システムのアクセス処理を行うアプリケーションプログラムのデータベースアクセス方式であって、

データベース管理システムの種別に依存しない形でインタフェース処理を行うアプリケーションプログラムインタフェースと、

複数の各々のデータベース管理システムに対してそれぞれにアクセス処理を行う各処理実行部と、

前記アプリケーションプログラムインターフェイスからの接続要求に基づいて操作対象のデータベース管理システムに対するアクセス処理を行う処理実行部とアプリケーションプログラムインターフェイスとの間の接続を行うサーバーとを備えることを特徴とするアプリケーションプログラムのデータベースアクセス方式。

【請求項２】　前記アプリケーションプログラムインターフェイスからの接続要求に基づいて、前記サーバーが処理実行部とアプリケーションプログラムインターフェイスとの間の接続を行う際に参照する各データベースに対する管理情報を格納したテーブルを更に備えることを特徴とする請求項１に記載のアプリケーションプログラムのデータベースアクセス方式。

【請求項３】　前記アプリケーションプログラムインタフェースには、共通データ型の定義情報を含み、

前記各処理実行部には、各データベース管理システムに固有のデータ型と共通データ型との変換を行う変換部を更に備えたことを特徴とする請求項１に記載のアプリケーションプログラムのデータベースアクセス方式。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、アプリケーションプログラムのデータベースアクセス方式に関し、さらに詳しくは、データベースを操作するデータベース管理システム（以下、ＤＢＭＳと略称する）へのアクセス処理を伴うアプリケーションプログラムを開発する場合において、そのアプリケーションプログラムにおけるデータベースアクセスモジュールの開発が効率よく行え、しかもアプリケーションプログラムにおけるデータベースのアクセス処理が能率よく行えるアプリケーションプログラムのデータベースアクセス方式に関するものである。

【０００２】

【従来の技術】従来、ＤＢＭＳを操作する処理を含むアプリケーションプログラムを開発するためには、アプリケーションプログラムの開発者が各々のＤＢＭＳのベンダーから提供されるＤＢＭＳアクセス用の開発支援ツールを利用するという形態で、ソフトウェア（アプリケーションプログラム）を開発している。この各々のデータベース提供者であるＤＢＭＳベンダーから提供される開発支援ツール（以下、ＤＢＭＳ用開発支援ツールと略称する）は、その各々のＤＢＭＳベンダーのＤＢＭＳをアクセスし、当該ＤＢＭＳベンダーから提供されるデータベースを利用するソフトウェアの開発の生産性の向上を目的としたものであり、いわゆるＤＢＭＳのアクセスモジュールとして提供されている。

【０００３】しかし、各々のＤＢＭＳ用開発支援ツールには、それぞれに固有の差異があるため、アプリケーションプログラムの開発者が複数のＤＢＭＳベンダーのＤＢＭＳのアクセス処理を含むプログラムを開発する場合に、利用対象とするＤＢＭＳによって、その対応のＤＢＭＳ用開発支援ツールに替えて、各々のＤＢＭＳ用開発支援ツールを個別に利用したアプリケーションプログラムの開発を行なわなければならない。

【０００４】このように、ＤＢＭＳを操作する処理を含むアプリケーションプログラムを開発する場合、各々のＤＢＭＳによって、それぞれにその対応のＤＢＭＳ用開発支援ツールに用いて、プログラム開発を行なわなければならないため、ソフトウェア作成作業の生産性の向上が図かれない。このため、プログラム開発の生産性を向上させるため、それぞれに異なるＤＢＭＳを意識させないアプリケーションプログラム開発方式が提案されている。例えば、特開平３－９０９３３号公報に記載されている「ソフトウェアプログラム自動生成方式」では、アクセスするファイルの属性情報に従い、そのＤＢＭＳに合った形のアクセス言語を自動生成する方式を提案している。

【０００５】特開平３－９０９３３号公報に記載されている「ソフトウェアプログラム自動生成方式」によれば、データベース処理を指定する機能を有するソフトウェアプログラムの開発を能率よく行うため、アクセスするファイルの属性情報から生成するデータベース言語の種別を判別し、該当のデータベース言語を自動生成する開発支援ツールが提供される。このため、このような開発支援ツールを用いることにより、利用者は、ＤＢＭＳの構造、種別の相違と、これに依存するデータベース言語の相違を意識せずに、プログラム開発を行うことができる。この開発支援ツールには、対象のデータベース表とＤＢＭＳ種別の対応テーブルが備えられており、ユーザが対象とするデータベースのファイル名を指示すると、開発支援ツールがアクセスするデータベースに対応して、データベース表からＤＢＭＳ種を判別し、そのＤＢＭＳに合った形のデータベース言語によるアクセス文を自動生成する。

【０００６】

【発明が解決しようとする課題】ところで、上述したように、従来におけるＤＢＭＳ用開発支援ツールでは、複数の異種ＤＢＭＳがネットワーク上で接続される分散型のデータベースシステムを構成するソフトウェア動作環境におけるアプリケーションプログラムを開発するためには、まず、各々のＤＢＭＳごとの開発支援ツールの利

用法を学習する必要があり、また、常に各々のＤＢＭＳを意識した上でのプログラム開発が必要となっている。そのため、データベースに対する操作の処理を伴うプログラム開発は、効率的に行うことはできず、ソフトウェア開発の生産性が向上していないという問題がある。

【０００７】これに対して、前述の特開平３－９０９３３号公報に記載の提案例に見られるように、各々に異なるＤＢＭＳを意識せずにアプリケーションプログラムが生成できる開発支援ツールの提案がなされており、このような開発支援ツールを用いてプログラム開発を行うことにより、ある程度に、ソフトウェアの生産性を向上させたプログラム開発も可能となっている。

【０００８】しかしながら、このような開発支援ツールを用い、個々のＤＢＭＳの特徴などを特に意識せず、異なる複数のＤＢＭＳに対するアクセス処理を含むアプリケーションプログラムを作成する場合、プログラム作成における生産性の向上は期待できるが、この開発支援ツールを用いて、複数のＤＢＭＳに対する操作を行うアプリケーションプログラムが作成されると、作成されたアプケーションプログラムは、複数のＤＢＭＳに対する個々のアクセスモジュールを、アプリケーションプログラムの側でそれぞれに持つ構造のプログラムが作成される。

【０００９】この場合、アプリケーションプログラムにおいて作成された各ＤＢＭＳ用アクセス用プログラムは、それぞれが各ＤＢＭＳに依存するプログラムであり、統一されたユーザインタフェースから各ＤＢＭＳアクセスを行う場合には、当該ユーザインタフェースと生成された各々のＤＢＭＳ用アクセス用のプログラムとをリンクする必要がある。また、複数のＤＢＭＳ用のアクセスモジュールをアプリケーションプログラムの側で持つことになるため、データベースの種類に応じて、操作対象のＤＢＭＳの数が増加すると、アクセス対象のＤＢＭＳおよびデータベース表の増加に伴い、アプリケーションプログラムの性能が落ちてしまう。更には、各ＤＢＭＳのデータ型の相違を吸収する部分がないため、ユーザインタフェースが同じ内容のプログラムとならないという問題がある。

【００１０】図２により具体的に説明すると、図２は、上述したようなＤＢＭＳ用開発支援ツールを用いて作成されたアプリケーションプログラムによるデータベースアクセス処理を説明する図である。図２において、１はデータベースに対するアクセス処理を伴うアプリケーションプログラムが走るワークステーション、２は第１のデータベース提供者のワークステーション、３は第２のデータベース提供者のワークステーション、４はローカルエリアネットワークを構成するネットワークである。１５はベンダ１のデータベース管理システム（ＤＢＭＳ１）、１６はベンダ１のデータベース本体部、１７はベンダ２のデータベース管理システム（ＤＢＭＳ２）、１８はベンダ２のデータベース本体部である。また、２０はアプリケーションプログラム本体部、２１はベンダー１のＤＢＭＳをアクセスするベンダー１用アクセスインタフェースであり、２２はベンダー２のＤＢＭＳをアクセスするベンダー２用アクセスインタフェースである。

【００１１】このようなシステム要素により、ソフトウェア動作環境が形成され、ここで動作するアプリケーションプログラム２０のソフトウェア動作環境は、複数の異種ＤＢＭＳがネットワーク上で接続される分散型のデータベースシステムとなっている。アプリケーションプログラム２０は、上述のような各ＤＢＭＳ用開発支援ツールにより生成されたものであり、ここでのアプリケーションプログラム２０には、２つの異なるＤＢベンダーのデータベースを操作するため、ベンダー１のＤＢＭＳをアクセスするベンダー１用アクセスインタフェース２１と、ベンダー２のＤＢＭＳをアクセスするベンダー２用アクセスインタフェースとが、それぞれ個別に設けられたプログラム構造で作成されたものとなっている。

【００１２】また、各々のＤＢＭＳで異なるデータ型の相違を吸収するためには、アプリケーションプログラム２０において、そのデータ型の解釈を行う処理プログラムを備える。そのため、アプリケーションプログラム２０は、ベンダー１のＤＢＭＳに対して、ベンダー１データ型解釈処理２０ａのプログラムが設けられ、ベンダー２のＤＢＭＳに対して、ベンダー２データ型解釈処理２０ｂのプログラムが設けられた構造のプログラムとなっている。これらのデータ型解釈を行う処理プログラムは、必要に応じてアプリケーションプログラム２０内に埋め込まれる。

【００１３】各ベンダー用のアクセスインタフェースにより、各々のＤＢＭＳをアクセスする場合、各ベンダー用のデータ型解釈処理部（２０ａ，２０ｂ）がそれぞれのアクセスインタフェースとリンクされ、アプリケーションプログラムにおけるデータ型の解釈を行い、アプリケーションプログラムのデータ型を各々のベンダー用のデータ型に合せて、アクセスインタフェース（２１，２２）が各々のＤＢＭＳに対するアクセス処理を行う。

【００１４】したがって、ここでのアプリケーションプログラム２０は、操作対象の各ベンダー用のアクセスインタフェース、各ベンダー用のデータ型解釈処理の処理プログラムを備える結果、プログラム自体が重くなり、全体的な性能が低下する。また、アクセスする対象のデータベースが増加した場合など、これらの各ベンダー用のＤＢＭＳのアクセスインタフェースに対応して装備するデータ型解釈処理のプログラムは、アプリケーションプログラムの開発者の側で作成しなければならず、将来のバージョンアップなどに容易に対応できないなど、プログラムの保守性が悪いという問題がある。

【００１５】本発明の上述のような問題点を解決するためになされたもので、本発明の目的は、データベースを

操作するＤＢＭＳへのアクセス処理を伴うアプリケーションプログラムを開発する場合において、そのアプリケーションプログラムにおけるデータベースアクセスモジュールの開発が効率よく行え、しかもシステム全体として各々のアプリケーションプログラムのデータベースのアクセス処理が能率よく行え、プログラムの保守性が向上したアプリケーションプログラムのデータベースアクセス方式を提供することにある。

【００１６】本発明の他の目的は、複数のＤＢＭＳへ同一のアプリケーションプログラムからのアクセスが共通のアクセスインタフェースで可能となるアプリケーションプログラムのデータベースアクセス方式を提供することにある。

【００１７】また、本発明の別の他の目的は、複数のＤＢＭＳへ同一のアプリケーションプログラムからのアクセスが可能となる開発支援ツール（共通ライブラリ）を提供することにより、アプリケーションプログラムの開発者に各ＤＢＭＡの構造や各アプリケーションインタフェースの違いを意識させず、更に、データ型の違いを意識させない（データ型の違いを吸収してしまう）構造のアプリケーションプログラムのデータベースアクセス方式を提供することにある。

【００１８】

【課題を解決するための手段】上述のような目的を達成するため、本発明のアプリケーションプログラムのデータベースアクセス方式は、アプリケーションプログラム（１０）においてデータベース管理システム（１５，１７）のアクセス処理を行うアプリケーションプログラムのデータベースアクセス方式であって、データベース管理システムの種別に依存しない形でインタフェース処理を行うアプリケーションプログラムインタフェース（１１）と、複数の各々のデータベース管理システムに対してそれぞれにアクセス処理を行う各処理実行部（１３）と、前記アプリケーションプログラムインターフェイスからの接続要求に基づいて操作対象のデータベース管理システムに対するアクセス処理を行う処理実行部とアプリケーションプログラムインターフェイスとの間の接続を行うサーバー（１２）とを備えることを特徴とする。

【００１９】また、ここでは、前記サーバー（１２）が、前記アプリケーションプログラムインターフェイスからの接続要求に基づいて、処理実行部とアプリケーションプログラムインターフェイスとの間の接続を行う際に、参照する各データベースに対する管理情報を格納したテーブル（１４）を更に備えることを特徴とする。

【００２０】また、前記アプリケーションプログラムインタフェース（１１）には、共通データ型の定義情報を含み、前記各処理実行部（１３）には、データベース管理システムに固有のデータ型と共通データ型との変換を行う変換部（６，８）を更に備えたことを特徴とする。

【００２１】

【作用】本発明のアプリケーションプログラムのデータベースアクセス方式によると、複数のデータベース管理システム（１５，１６）のアクセス処理を行うアプリケーションプログラム（１０）には、データベース管理システムの種別に依存しない形で、各処理実行部とのインタフェース処理を行うアプリケーションプログラムインタフェース（１１）が設けられ、アプリケーションプログラム（１０）が動作するソフトウェア動作環境を提供するシステム内に、各アプリケーションプログラムが共通に使用できる各々のデータベース管理システムとの間のアクセス処理を行う各処理実行部（１３）と、サーバー（１２）とが備えられる。

【００２２】アプリケーションプログラム（１０）に備えられたアプリケーションプログラムインタフェース（１１）は、データベース管理システムの種別に依存しないインタフェース処理を行い、各処理実行部（１３）が複数の各々のデータベース管理システムに対してそれぞれのアクセス処理（５，７）を行う。また、サーバー（１２）は、アプリケーションプログラムインターフェイス（１１）からの接続要求に基づいて、対象のデータベース管理システムに対するアクセス処理を行う個別の処理実行部（１３ａ，１３ｂ）と接続要求を出したアプリケーションプログラムインターフェイス（１１）との間の接続を行う。

【００２３】このため、サーバーが処理実行部とアプリケーションプログラムとの接続を行う際に参照する各データベースに対応する管理情報を格納したテーブル（１４）が備えられおり、サーバー（１２）は、テーブル（１４）の内容を参照して、サーバーが処理実行部とアプリケーションプログラムインターフェイスとの間の接続を行う。

【００２４】アプリケーションプログラム（１０）においては、アプリケーションプログラムインタフェース（１１）が、複数のデータベース管理システムに対してその種別に依存しないインタフェース処理を行うが、この際、操作者に対して共通の操作によるデータベース操作の処理が可能なように、データ型の相違に対する制約を吸収して、アプリケーションプログラムインタフェース（１１）は共通データ型でのインタフェース処理を行う。このため、アプリケーションプログラムインタフェース（１１）には共通データ型の定義情報を含んでおり、この定義情報を利用してデータ型変換を行う変換部（６，８）が、更に各処理実行部（１４）に備えられている。変換部（６，８）はデータベース管理システムに固有のデータ型と共通データ型との変換を行う。この定義情報による共通データ型でアプリケーションプログラムインタフェース（１１）と各処理実行部（１４）との間の連絡を行い、その後、データ型変換を行い、各処理実行部（１４）が、各データベース管理システムとの間でそれぞれのデータベース管理システムの固有のデータ

型で連絡を行う。

【００２５】したがって、変換部（６，８）により、各々のデータベース管理システムの間のデータ型はその相違が吸収されるので、アプリケーションプログラムの開発者は、特に、各ＤＢＭＡの構造や各アプリケーションプログラムインタフェースの違い、更には、データ型の違いを意識せず、アプリケーションプログラムの開発を行うことができ、その開発負担が軽減され、開発の生産性を向上させることができる。

【００２６】このように、アプリケーションプログラムにおいてデータベースアクセス処理を行う場合、サーバーがアプリケーションプログラムインターフェイスからの接続要求に基づいて、対象のＤＢＭＳに対するアクセス処理を行う処理実行部との間の接続を行い、アプリケーションプログラムインタフェースおよび処理実行部を介して、対象となる１つのＤＢＭＳベンダーの１つのデータベースに対するアクセス処理を行う。アクセスする対象のデータベースが異なる場合には、そのＤＢＭＳに対応する処理実行部への接続の切換えをサーバーが行い、アプリケーションプログラムからは共通のアプリケーションプログラムインタフェースを介して、データベースに対するアクセス処理が行なわれる。

【００２７】ここでは、アプリケーションプログラム側には、アプリケーションプログラムインタフェースのみが備えられており、アプリケーションプログラムにおけるデータベースに対するアクセス処理は、アプリケーションプログラムインタフェース，各処理実行部，およびサーバーを用いて行なわれる。このため、データベースに対する操作を含むアプリケーションプログラムを開発する場合、特に、アプリケーションプログラムには、各々のＤＢＭＳの操作に共通なアプリケーションプログラムインタフェースのみを備えていれば良いので、アプリケーションプログラム開発者に各ＤＢＭＳの構造や各ＤＢＭＳ用のアプリケーションプログラムインタフェースを意識しないで済む。

【００２８】これにより、アプリケーションプログラム開発者は、各ＤＢＭＳの構造や各ＤＢＭＳ用のアプリケーションプログラムインタフェースを意識しないで済み、また、データ型の相違も意識しないで済むので、アプリケーションプログラムの開発者の負担が軽減される。このため、ソフトウェア開発の生産性を向上させることができる。また、アプリケーションプログラムが複数のデータベースに対する操作を行う場合にも、アプリケーションプログラムインタフェースがサーバーに対するデータベースへの接続要求のみを発行すればよいので、アプリケーションプログラム自体が重くなり、性能が落ちてしまうことはない。更にアクセスする対象のデータベースが新たに増加した場合には、新たに増加したＤＢＭＳに対するＤＢＭＳ用の処理実行部のみを、ＤＢＭＳベンダーから提供される開発支援ツールを用いて作成し、必要であればデータ型変換部を含んだ処理実行部を作成して、システム内にインストールするだけで良く、格別にアプリケーションプログラムの修正作業を行うことなく対応できる。このため、将来のバージョンアップなどに容易に対応でき、プログラムの保守性が良い。

【００２９】

【実施例】以下、本発明の一実施例を図面を参照して具体的に説明する。図１は、本発明の一実施例にかかるアプリケーションプログラムのデータベースアクセス方式の要部の構成を示すブロック図である。図１において、１はデータベースのアクセス処理を伴うアプリケーションプログラムが走るワークステーション、２は第１のデータベース提供者のワークステーション、３は第２のデータベース提供者のワークステーション、４はローカルエリアネットワークを構成するネットワークである。また、１０はアプリケーションプログラム、１１はアプリケーションプログラムインタフェース、１２はサーバー、１３（１３ａ，１３ｂ）は各々のＤＢＭＳベンダ用処理実行部、１４は処理実行部エントリテーブルをそれぞれ示している。１５はベンダ１のデータベース管理システム（ＤＢＭＳ）、１６はベンダ１のデータベース本体部、１７はベンダ２のデータベース管理システム（ＤＢＭＳ）、１８はベンダ２のデータベース本体部である。なお、各々のＤＢＭＳベンダ用処理実行部１３は各ベンダー向けアクセス処理（１３ａ，１３ｂ）に対応して、ＤＢＭＳ１アクセス処理実行部５，ＤＢＭＳ１／共通データ型変換処理部６，ＤＢＭＳ２アクセス処理実行部７，ＤＢＭＳ２／共通データ型変換処理部８をそれぞれ備えている。

【００３０】ここでのアプリケーションプログラム１０が走るソフトウェア動作環境は、図１に示すように、各ワークスーション１，２，３がネットワーク４により相互に接続されたネットワークシステムとなっており、ワークステーション２，３で複数の異種ＤＢＭＳがネットワーク上に接続された分散型のデータベースシステムを構成している。アプリケーションプログラム１０は、２つの異なるＤＢベンダーのＤＢＭＳにおけるデータベースを操作する場合にも、それぞれに対応したアクセスインタフェースは必要とせず、共通に用いるアプリケーションプログラムインタフェース１１のみで良い。また、データベース処理では、共通データ型でデータベース操作を行うため、共通データ型解釈処理を行い、アプリケーションプログラムインタフェース１１とのリンクを行う。データベースのアクセス処理を伴うアプリケーションプログラム１０が走るワークステーション１には、図示するように、サーバー１２，各々のＤＢＭＳベンダ用処理実行部１３，および処理実行部エントリテーブル１４が備えられている。

【００３１】ＤＢＭＳベンダ用処理実行部１３には、各

々のＤＢＭＳに対するアクセス処理を行うための処理実行部として、第１のＤＢＭＳベンダ用処理実行部５，第１のＤＢＭＳ用データ型と共通データ型との間のデータ型変換を行うＤＢＭＳ１／共通データ型変換処理部６，第２のＤＢＭＳベンダ用処理実行部７，第２のＤＢＭＳ用データ型と共通データ型との間のデータ型変換を行うＤＢＭＳ２／共通データ型変換処理部８が備えられている。ここでの共通データ型の定義情報は、アプリケーションプログラムインタフェース１１に含まれており、必要に応じて参照される。

【００３２】なお、データベースに対するアクセス処理を伴うアプリケーションプログラム１０には、データベースのアクセスのために、アプリケーションプログラムインタフェース１１のみが付加された形態でのプログラム開発が行なわれ、サーバー１２，各々のＤＢＭＳベンダ用処理実行部１３（５，６，７，８）および処理実行部エントリテーブル１４が備えられたワークステーション１の上で、当該アプリケーションプログラム１０が走ることになる。

【００３３】アプリケーションプログラムインターフェイス１１は、アプリケーションプログラム開発において、ＤＢＭＳ非依存のインタフェース機能を提供しているプログラムモジュールであり、プログラム開発者は、この共通のインタフェース機能を習熟することにより、複数のＤＢＭＳを扱う場合にも、各ＤＢＭＳの構造や各ＤＢＭＳ用アプリケーションプログラムインタフェース，データ型を意識することなく、プログラム開発を行うことができる。

【００３４】また、サーバー１２は、当該ワークステーション１において、アプリケーションプログラムインタフェース１１からの接続要求を見張り、その接続要求が発行された場合に、対象となる各ＤＢＭＳベンダ用処理実行部１３における各々のアクセス処理部（１３ａ，１３ｂ）との接続サービスを行う機能を提供する。ここでは、データ型の相違を吸収するため、アプリケーションプログラムインタフェース１１と各々のＤＢＭＳアクセス処理実行部（５，７）との間の接続は、各々のＤＢＭＳ／共通データ型変換処理部（６，８）を介して行う。

【００３５】ＤＭＢＳベンダ用処理実行部１３の各々のＤＢＭＳアクセス処理実行部は、各ベンダーからの開発支援ツールにより各ベンダーＤＢＭＳ用アプリケーションインタフェースを利用して作成されており、当該ワークステーション１において、各ＤＢＭＳのアクセスメソッドの相違，データ型の相違を吸収して、データベースを操作する対象のＤＢＭＳに対するアクセス処理（発行，受取り）機能を提供する。

【００３６】また、処理実行部エントリテーブル１４は、サーバー１２がアプリケーションプログラム１０からのデータベース接続要求に対して接続処理を行う場合に参照される管理情報が登録されているテーブルであり、接続要求されるＤＢＭＳ名に対して、ＤＢＭＳベンダ用処理実行部名を対応させて登録してある。アプリケーションプログラム１０における処理の中からの接続要求に対して、ＤＢＭＳベンダ用処理実行部１３の何れのＤＢＭＳのアクセス処理部（１３ａ，１３ｂ）との接続を行うかの対応が参照される。

【００３７】図３は処理実行部エントリテーブルの構成例を示す図である。処理実行部エントリテーブル１４は、具体的に説明すると、図３に示すように、ＤＢＭＳの名前を登録するＤＢＭＳ名フィールド３１と、対応ＤＢＭＳのアクセスツールを登録するＤＢＭＳアクセス処理実行部名フィールド３２とから構成されており、ＤＢＭＳ名に対応して起動すべきＤＢＭＳアクセス処理実行部の名前を登録してあるエントリテーブルである。また、この分散型データベースシステムにおいて接続可能な各々のＤＢＭＳを管理している管理情報となっている。前述したように、アプリケーションプログラムインタフェース１１からＤＢＭＳ名を指定した接続要求が発行された場合、サーバー１２が、処理実行部エントリテーブル１４を参照して、ＤＢＭＳ名から対応するＤＢＭＳアクセス処理実行部プログラム名を知り、該当のＤＢＭＳアクセス処理実行部プログラムを起動する。

【００３８】次に、このような動作環境で動作するアプリケーションプログラム１０におけるデータベースに対するアクセス処理の概略を説明する。まず、アプリケーションプログラム１０の処理において、データベース操作での共通データ型の解釈処理が行なわれ、ＤＢＭＳのアクセス処理を行うステップに処理が進行すると、アプリケーションプログラムインタフェース１１が、アプリケーションプログラム本体部分とのインタフェース処理により、サーバー１２に対してＤＢＭＳに接続要求を発行する。サーバー１２は、アプリケーションプログラムインターフェイス１１からの接続要求に基づいて、処理実行部エントリテーブル１４における対応情報を参照して、対象のデータベース管理システムに対するアクセス処理を行うＤＢＭＳベンタ用処理実行部１３の対応のアクセス処理部を起動し、次にアプリケーションプログラムインタフェース１１を介してアプリケーションプログラム１０の本体部との間の接続を行う。これにより、接続されたＤＢＭＳベンタ用処理実行部１３の例えばＤＢＭＳアクセス処理部１３ｂは、操作対象のＤＢＭＳに対してのアクセス処理を行う。

【００３９】このように、アプリケーションプログラム１０においてデータベースアクセスを行う場合、サーバー１２がアプリケーションプログラムインターフェイス１１からの接続要求に基づいて、対象のＤＢＭＳに対するアクセス処理を行う個別ＤＢＭＳベンダ用処理実行部（１３ａまたは１３ｂ）を起動し、例えばＤＢＭＳベンダ用アクセス処理実行部１３ｂとアプリケーションプログラムインターフェイス１１との間の接続を行い、ＤＢ

ＭＳベンダ用処理実行部１３が対象のＤＢＭＳとの間のアクセス処理を行ってその間の接続を行う。したがって、データベースアクセス処理では、アプリケーションプログラムインタフェース１１およびＤＢＭＳベンダ用処理実行部１３を介して、対象となるＤＢＭＳのデータベース操作を行うことになる。

【００４０】次に、図４～図７を参照して、各々の処理モジュールの間の接続を行い、順次にデータベースに対するアクセス処理を進んで行く様子の各状態を説明する。図４は、アプリケーションプログラムが実行された直後の時点の状態を示す図であり、図５は、アプリケーションプログラムがアプリケーションプログラムインターフェイスの接続要求を行った時点の状態を示す図である。図６は、サーバーが操作対象のＤＢＭＳベンダ用処理実行部を起動した時点の状態を示す図であり、また、図７は、アプリケーションプログラムがアプリケーションプログラムインターフェイスを通してアクセス要求を行った場合の状態を示す図である。

【００４１】まず、図４に示すような初期状態から、アプリケーションプログラム１０が、各ベンダから提供されているデータベースへのアクセス動作を行う場合、図５に示すように、まず、アプリケーションプログラム１０がアプリケーションプログラムインターフェイス１１を介してサーバー１２に接続要求①を発行する。接続要求①が発行されると、サーバー１２は、この接続要求①をキャッチする。サーバー１２は接続要求①の要求パラメータ（どのＤＢＭＳに接続するのか）を判別し、その内容から処理実行部エントリテーブル１４を参照して、対象のＤＢＭＳに対してアクセス処理を実行するＤＢＭＳベンダ用処理実行部プログラム名を確定する処理②を行う。

【００４２】次に、図６に示すような状態となり、サーバ１２が処理実行部エントリテーブル１４により確定されたプログラム名から当該するＤＢＭＳ２アクセス処理実行部７を起動③する。起動されたＤＢＭＳ２アクセス処理実行部７は、アプリケーションプログラム１０のアプリケーションプログラムインタフェース１１との間の接続を行うため、対応のＤＢＭＳ２のデータ型と共通データ型との間のデータ型変換を行うＤＢＭＳ２／共通データ型変換処理部８を起動し、このＤＢＭＳ２／共通データ型変換処理部８を介して、この間のデータベースアクセス処理を行うパス④を確立する。この時点でサーバー１２は、他のアプリケーションプログラムがそのアプリケーションプログラムインタフェースを通して要求をしてくる次の接続要求の発行を待つ状態となる。

【００４３】そして、次の状態は図７に示すような状態となり、起動されたＤＢＭＳ２アクセス処理実行部７は、アプリケーションプログラムインタフェース１１からの要求により対象とするＤＢＭＳ２（１７）との接続処理を行い、その間のデータベースアクセス処理を行うパス⑤を確立する。この結果、先に確立されたアプリケーションプログラムインタフェース１１とのアクセスパス④およびその後に確立されたＤＢＭＳ２（１７）とのアクセスパス⑤を通して、アプリケーションプログラム１０がアプリケーションプログラムインタフェース１１およびＤＢＭＳ２ベンダ用処理実行部１３ｂの各処理モジュール（ＤＢＭＳ２アクセス処理実行部７、ＤＢＭＳ２／共通データ型変換処理部８）を介して、操作対象のデータベース本体部１８のＤＢＭＳ２（１７）に対してアクセス処理を行う状態となる。ここでのＤＢＭＳ２（１７）に対するアクセス処理が終了すると、その後、アプリケーションプログラム１０がアプリケーションプログラムインタフェース１１を通して、切断要求を行った後、ＤＢＭＳ２アクセス処理実行部７が接断処理を行い、処理を終了して、図４に示すような状態に戻る。

【００４４】次に、図８～図１１を参照して、アプリケーションプログラムからのデータベースのアクセス処理において各々のベンダーのＤＢＭＳ固有のデータ型の違いがどのように吸収されるかを、前述の説明と同様に順を追って説明する。図８は、アプリケーションプログラムから検索命令を発行した時の状態を示す図であり、図９は、ＤＢＭＳアクセス処理実行部から操作対象のＤＢＭＳに対して検索命令を発行する時の状態を示図である。図１０は、検索命令を発行したＤＢＭＳアクセス処理実行部に操作対象のＤＢＭＳからの検索結果が返ってきた時の状態を示す図であり、また、図１１は、ＤＢＭＳアクセス処理実行部が検索命令を発行したアプリケーションプログラムに検索結果を返す時の状態を示す図である。

【００４５】ここでは、データベースに対してアプリケーションプログラムから検索命令を発行する場合を例にして具体的に説明する。まず、図８に示すように、アプリケーションプログラム１０において発行されるデータベースの検索命令４１は、アプリケーションプログラムインタフェース１１から発行されるが、この場合、検索結果は共通データ型で受け取ることを指定して、ＤＢＭＳ２のアクセス処理を行うＤＢＭＳ２アクセス処理部１３ｂに対して検索命令４１を発行する。すなわち、アプリケーションプログララム１０では、検索命令４１のパラメータで、アプリケーションプログラムインタフェース１１において自分の受け取る検索結果のデータ型を定義情報で定義されている共通データ型から、例えば共通の文字列型とういうように指定して検索命令を発行する。

【００４６】ＤＢＭＳ２アクセス処理部１３ｂは、先の接続要求によりアクセス対象のＤＢＭＳ２に対するアクセス実行処理を行うＤＢＭＳ２アクセス処理実行部７およびＤＢＭＳ２／共通データ型変換処理部８が起動されており、この共通データ型による検索命令４１を、ＤＢＭＳ２アクセス処理実行部７はＤＢＭＳ２／共通データ

型変換処理部８を介して、共通データ型からベンダー２用データ型の検索命令に変換して受け付ける。

【００４７】次に、ベンダー２用データ型の検索命令に変換して受け付けたＤＢＭＳ２アクセス処理実行部７は、データベース管理システム（ＤＢＭＳ２）１７に対するアクセス処理を行い、図９に示すように、ベンダー２用データ型の検索命令４２をデータベース管理システム（ＤＢＭＳ２）１７に送信する。これにより、データベース管理システム（ＤＢＭＳ２）１７においては、検索命令４２によりベンダー２データベース本体部１８に対して、データベース検索処理が行なれ、その検索結果が返される。

【００４８】この結果、図１０に示すように、検索結果は、データベース管理システム（ＤＢＭＳ２）１７からベンダー２用データ型で検索結果４３が返される。ベンダー２用データ型での検索結果４３は、ＤＢＭＳ２アクセス処理実行部７において受け取られる。次に、ＤＢＭＳ２アクセス処理実行部７は、図１１に示すように、ベンダー２用データ型での検索結果４３をＤＢＭＳ２／共通データ型変換処理部８を介して共通データ型の検索結果へのデータ型変換を行い、共通データ型の検索結果４４として、アプリケーションプログラムインタフェース１１に検索結果を返す。これにより、アプリケーションプログラム１０においては、共通データ型の検索結果を利用する。このようにして、データベースの検索処理における各々のデータ型の変換が行なわれる。また、データベースに対して入力操作を行う場合、更新操作を行う場合には、共通データ型から各ベンダーのＤＢＭＳに対応するデータ型への変換が行なわれて、処理される。

【００４９】図１２はアプリケーショプログラムインタフェースに設けられる共通データ型の定義情報の構成例を示す図である。この定義情報は、図１２に示すように、共通データ型５１の定義情報と、この共通データ型５１と対応する各々の各ベンダーのＤＢＭＳのデータ型との対応テーブルとして定義されている。ここでは共通データ型５１であるＲＣＡデータ型と、このシステムでアクセス処理が可能となっている第１のＤＢＭＳ１の「ＯＲＡＣＬＥ」のデータ型５２、第２のＤＢＭＳ２の「ｉｎｆｏｒｍｉｘ」のデータ型５３、および第３のＤＢＭＳ３の「Ｓｙｂａｓｅ」のデータ型５４が、対応づけられている。これらのデータ型の変換は、１対１に対応づけられるものでなく、アプケーションプログラムの内容に応じて任意に対応づけられる。

【００５０】図１３、図１４、および図１５は、検索処理において各々のＤＢＭＳの固有のデータ型がアプリケーションプログラムインタフェースの共通データ型に変換される場合の対応関係の具体例を示す図である。図１３に、データベース「ＯＲＡＣＬＥ」のデータ型から共通データ型への変換関係を示し、図１４には、データベース「ｉｎｆｏｒｍｉｘ」のデータ型から共通データ型への変換関係を示している。また、図１５に、データベース「Ｓｙｂａｓｅ」のデータ型から共通データ型への変換関係を示している。これらの図の変換関係から明らかなように、本実施例のアプリケーションプログラムのデータベースアクセス方式を用いることにより、各々のデータベースに固有の複雑なデータ型の指定を、共通データ型により指定で容易に行うことができる。

【００５１】以上に説明したように、アプリケーションプログラム１０がデータベースの操作を伴うアクセス処理を行う場合、アプリケーションプログラムインターフェイス１１からの接続要求に基づいて、サーバー１２が対象のＤＢＭＳに対するアクセス処理を行うＤＢＭＳベンダ用処理実行部１３との間の接続を行い、アプリケーションプログラムインタフェース１１およびＤＢＭＳベンダ用処理実行部１３の各々の処理モジュールを介して、例えば操作対象となる１つのデータベース本体部１８に対するアクセス処理を行う。この際に各々のＤＢＭＳベンダ用処理実行部１３ではデータ型の相違も吸収する。

【００５２】各々のアプリケーションプログラム１０でアクセスする対象のデータベースが異なる場合には、そのＤＢＭＳに対応するＤＢＭＳベンダ用処理実行部１３への接続をサーバー１２が行い、アプリケーションプログラムからは共通仕様のアプリケーションプログラムインタフェース１１を介して、データベースに対するアクセス処理が行なう。このため、サーバー１２と共に備えられ、アプリケーションプログラムインタフェースからの接続要求を受けて、サーバーがＤＢＭＳベンダ用処理実行部１３とアプリケーションプログラムインタフェース１１との間の接続を行うための情報を与える処理実行部エントリテーブル１４が設けられる。

【００５３】以上に説明した本実施例による効果をまとめると次のようになる。

（１）従来においては、各ベンダー用ＤＢＭＳのアプリケーションプログラムインタフェースが異なるため、アプリケーションプログラムもそれに依存したものとなり、個別にアプリケーションプログラムを開発しなければならなかったが、本実施例のようなアクセス方式によると、複数のＤＢＭＳにアクセスするアプリケーションプログラムが同一のものですむ。すなわち、アプリケーションプログラムはアプリケーションプログラムインタフェースにより、ＤＢＭＳに対する接続要求を発行する際に、ＤＢＭＳを指定するだけでよい。また、各ＤＢＭＳ用のデータ型をアプリケーションプログラムの処理でそれぞれ解釈する必要があったが、共通データ型を用いることにより、共通データ型の解釈を行うだけで、各々のＤＢＭＳ用のデータ型を扱うことが可能になる。

【００５４】（２）従来においては、各ベンダＤＢＭＳ用開発援ツールにおけるアプリケーションプログラムインタフェース仕様を修得しなければ、各ＤＢＭＳ用アプ

リケーションプログラムの開発は不可能であったが、本実施例によるアクセス方式を用いることにより、アプリケーションプログラム開発者は、共通仕様のアプリケーションプログラムインタフェースの利用方法を学習するだけで、各ベンダーのＤＢＭＳ用アプリケーションの開発が可能となる。このため、ソフトウェア開発効率が向上する。言い換えると、ひとつのアプリケーションプログラムを作成すれば、それはベンダー１のＤＢＭＳにも、ベンダー２のＤＢＭＳにも、どちらにもアクセスできるアクセスインタフェースを備えていることになる。

【００５５】（３）将来、アプリケーションプログラムに対して、新たなＤＢＭＳ対応のバージョンアップを行う場合、その対応のベンダーＤＢＭＳ用処理実行部として、ＤＢＭＳアクセス処理実行部および当該ベンダー用ＤＢＭＳデータ型と共通データ型との間のデータ型変換処理部を追加するだけで良く、アプリケーションプログラム自体には手を加える必要はない。新しいベンダー用ＤＢＭＳ処理実行部が作成された時点で、処理実行部エントリテーブルに新しいベンダーのＤＢＭＳ名，ＤＢＭＳアクセス処理実行部名を追加すると、新たなＤＢＭＳに対するアクセスが可能となる。

【００５６】

【発明の効果】以上、説明したように、本発明のアプリケーションプログラムのデータベースアクセス方式によれば、各々のアプリケーションプログラムの側には、基本的にはアプリケーションプログラムインタフェースのみが備えられれば良いので、データベースに対する操作を含むアプリケーションプログラムを開発する場合、アプリケーションプログラム開発者に各ＤＢＭＳの構造や各ＤＢＭＳ用アプリケーションプログラムインタフェースを意識しないで済む。また、共通データ型へのデータ型変換処理部が設けられのので、データ型の相違を意識しないで済む。

【００５７】したがって、アプリケーションプログラム開発者は、特に、各ＤＢＭＳの構造や各ＤＢＭＳ用アプリケーションプログラムインタフェースを意識することはなく、データベースのアクセス処理を含むプログラムの開発を行える。アプリケーションプログラム開発者はその負担が軽減され、開発の生産性を向上させることができる。なお、アプリケーションプログラムが複数のデータベースに対する操作を行う場合、アプリケーションプログラムインタフェースがサーバーに対するデータベースへの接続要求のみを発行すればよいので、アプリケーションプログラムの性能が落ちてしまうことはない。

【図面の簡単な説明】

【図１】　図１は本発明の一実施例にかかるアプリケーションプログラムのデータベースアクセス方式の要部の構成を示すブロック図、

【図２】　図２は、従来におけるＤＢＭＳ用開発支援ツールを用いて作成されたアプリケーションプログラムによるデータベースアクセス処理の流れを説明する図、

【図３】　図３は処理実行部エントリテーブルの構成例を示す図、

【図４】　図４～図７は本実施例によるアクセス処理の各状態を説明する図であり、図４はアプリケーションプログラムが実行された直後の時点の状態を示す図、

【図５】　同じく、図５はアプリケーションプログラムがアプリケーションプログラムインターフェイスの接続要求を行った時点の状態を示す図、

【図６】　同じく、図６はサーバーが操作対象のＤＢＭＳベンダ用処理実行部を起動した時点の状態を示す図、

【図７】　同じく、図７はアプリケーションプログラムがアプリケーションプログラムインターフェイスを通してアクセス要求を行った場合の状態を示す図である。

【図８】　図８はアプリケーションプログラムからＤＢＭＳアクセス処理実行部に検索命令を発行した時の状態を示す図、

【図９】　図９はＤＢＭＳアクセス処理実行部から操作対象のＤＢＭＳに対して検索命令を発行する時の状態を示図、

【図１０】　図１０は検索命令を発行したＤＢＭＳアクセス処理実行部に操作対象のＤＢＭＳからの検索結果が返ってきた時の状態を示す図、

【図１１】　図１１はＤＢＭＳアクセス処理実行部が検索命令を発行したアプリケーションプログラムに検索結果を返す時の状態を示す図、

【図１２】　図１２はアプリケーショプログラムインタフェースに設けられる共通データ型の定義情報の構成例を示す図、

【図１３】　図１３は検索処理においてデータベース「ＯＲＡＣＬＥ」のデータ型からアプリケーションプログラムインタフェースの共通データ型に変換される場合の対応関係の具体例を示す図、

【図１４】　図１４は検索処理においてデータベース「ｉｎｆｏｒｍｉｘ」のデータ型からアプリケーションプログラムインタフェースの共通データ型に変換される場合の対応関係の具体例を示す図、

【図１５】　図１５は検索処理においてデータベース「Ｓｙｂａｓｅ」のデータ型からアプリケーションプログラムインタフェースの共通データ型に変換される場合の対応関係の具体例を示す図である。

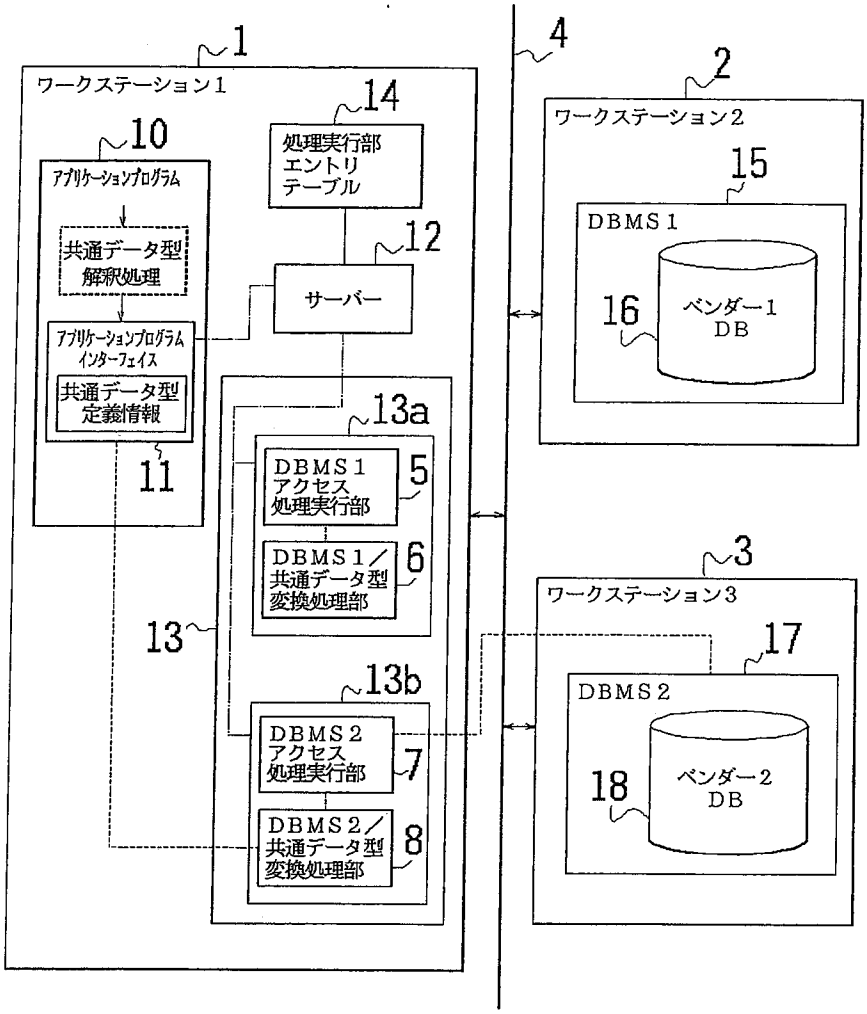
【符号の説明】

１…ワークステーション、２…ワークステーション、３…ワークステーション、４…ネットワーク、５…ＤＢＭＳ１アクセス処理実行部、６…ＤＢＭＳ１／共通データ型変換処理部、７…ＤＢＭＳ２アクセス処理実行部、８…ＤＢＭＳ２／共通データ型変換処理部、１０…アプリケーションプログラム、１１…アプリケーションプログラムインタフェース、１２…サーバー、１３…ＤＢＭＳベンダ用処理実行部、１４…処理実行部エントリテーブ

ル、１５…データベース管理システム（ＤＢＭＳ１）、１６…データベース本体部、１７…データベース管理システム（ＤＢＭＳ２）、１８…データベース本体部、２０…アプリケーションプログラム、２１…ベンダー１用アクセスインタフェース、２２…ベンダー２用アクセスインタフェース、３１…データベース名フィールド、３２…ＤＢＭＳアクセス処理実行部プログラム名フィールド。

【図１】

図１



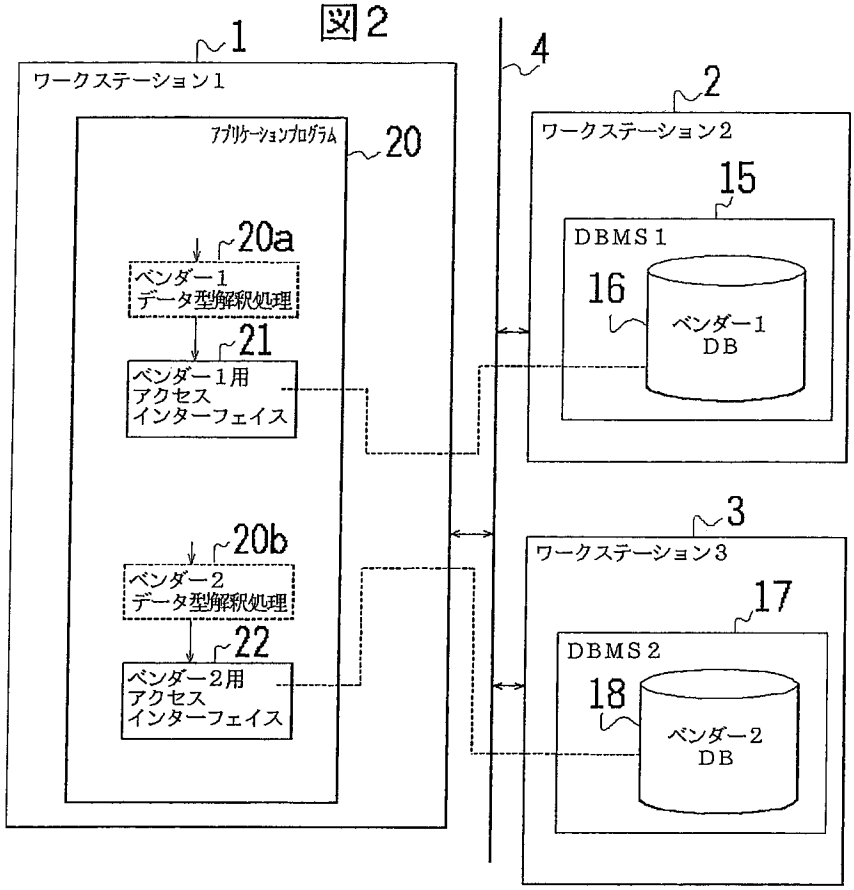
【図３】

図３

31 DBMS名	32 DBMSアクセス処理実行部名
ベンダー1　DBMS	ベンダー1用処理実行部プログラム名
ベンダー2　DBMS	ベンダー2用処理実行部プログラム名

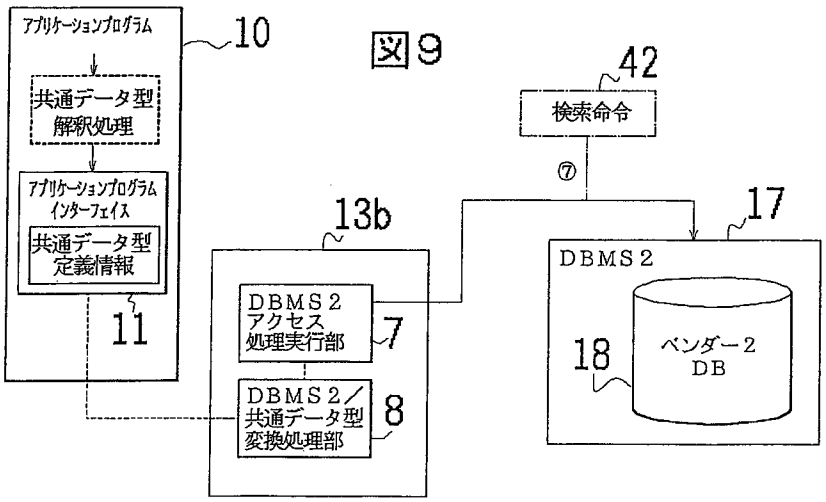
【図２】

図２


1
ワークステーション１
アプリケーションプログラム
20
20a
ベンダー１
データ型解釈処理
21
ベンダー１用
アクセス
インターフェイス
20b
ベンダー２
データ型解釈処理
22
ベンダー２用
アクセス
インターフェイス
4
2
ワークステーション２
15
DBMS１
16
ベンダー１
DB
3
ワークステーション３
17
DBMS２
18
ベンダー２
DB

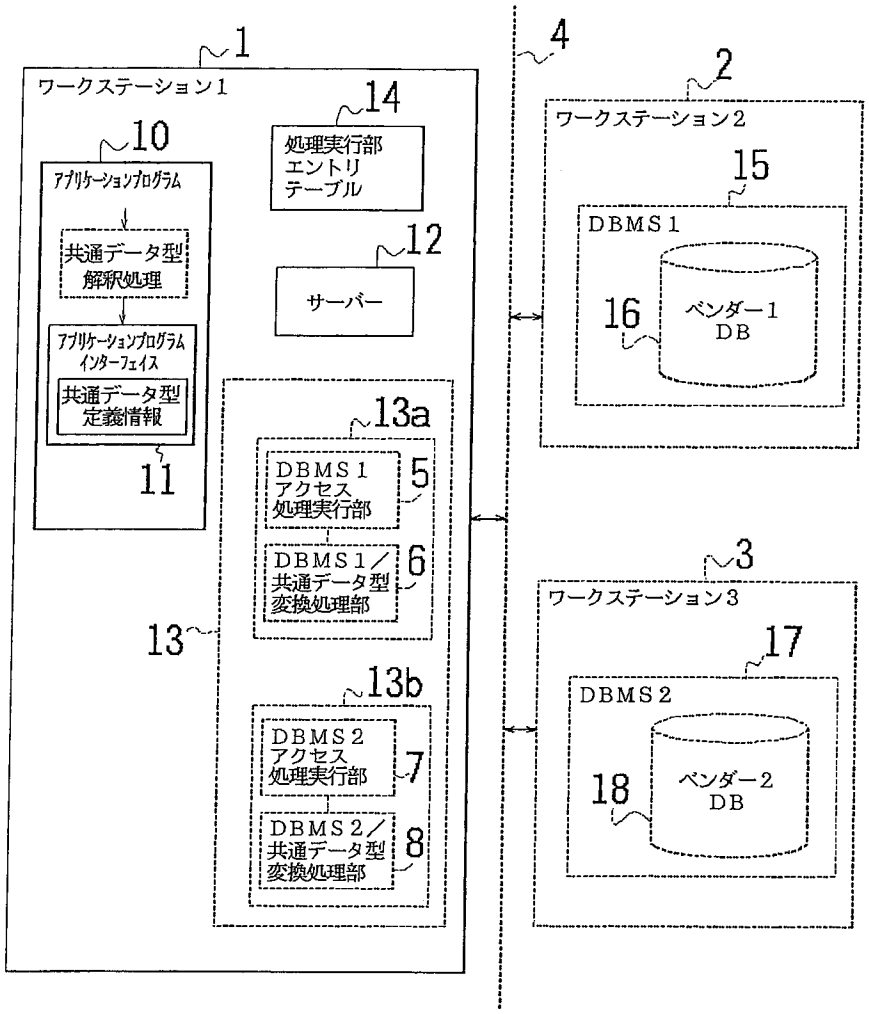
【図９】

図９


アプリケーションプログラム
10
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
42
検索命令
⑦
13b
DBMS２
アクセス
処理実行部
7
DBMS２／
共通データ型
変換処理部
8
17
DBMS２
18
ベンダー２
DB

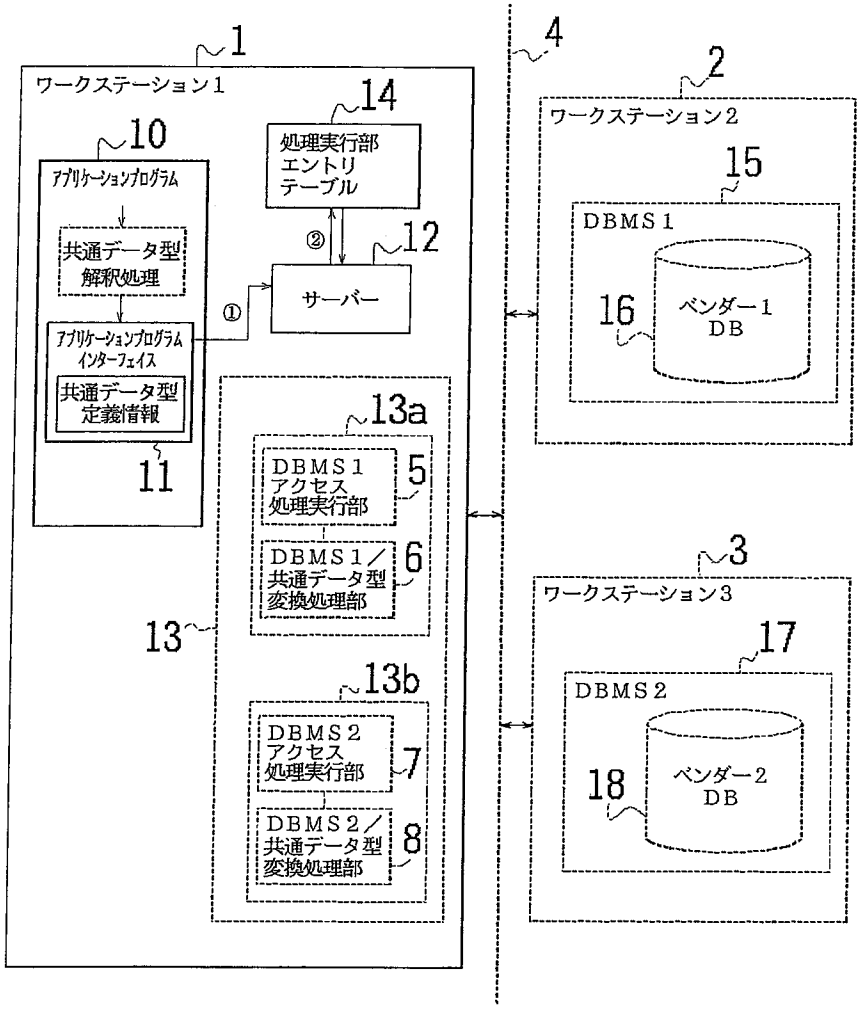
【図４】

図4


1
ワークステーション1
14
処理実行部
エントリ
テーブル
10
アプリケーションプログラム
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
12
サーバー
13a
DBMS1
アクセス
処理実行部
5
DBMS1／
共通データ型
変換処理部
6
13
13b
DBMS2
アクセス
処理実行部
7
DBMS2／
共通データ型
変換処理部
8
4
2
ワークステーション2
15
DBMS1
16
ベンダー1
DB
3
ワークステーション3
17
DBMS2
18
ベンダー2
DB

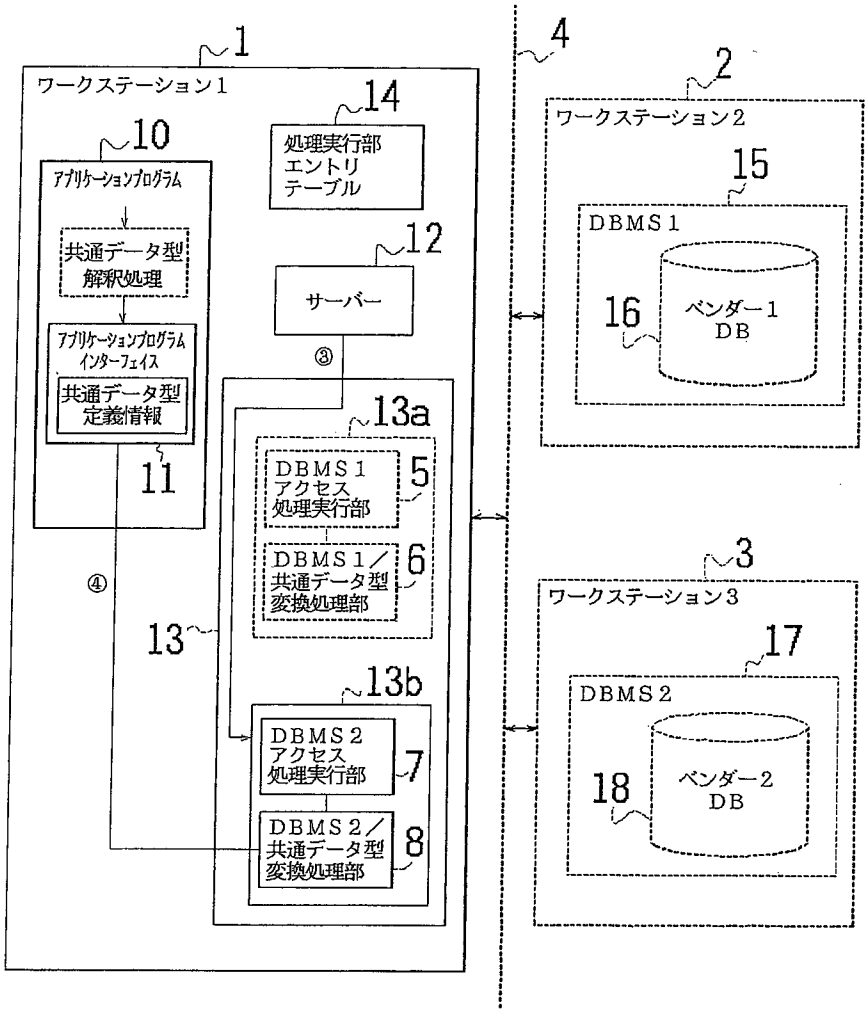
【図５】

図5


1
ワークステーション1
14
処理実行部
エントリ
テーブル
10
アプリケーションプログラム
共通データ型
解釈処理
②
12
サーバー
①
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
13a
DBMS1
アクセス
処理実行部
5
DBMS1／
共通データ型
変換処理部
6
13
13b
DBMS2
アクセス
処理実行部
7
DBMS2／
共通データ型
変換処理部
8
4
2
ワークステーション2
15
DBMS1
16
ベンダー1
DB
3
ワークステーション3
17
DBMS2
18
ベンダー2
DB

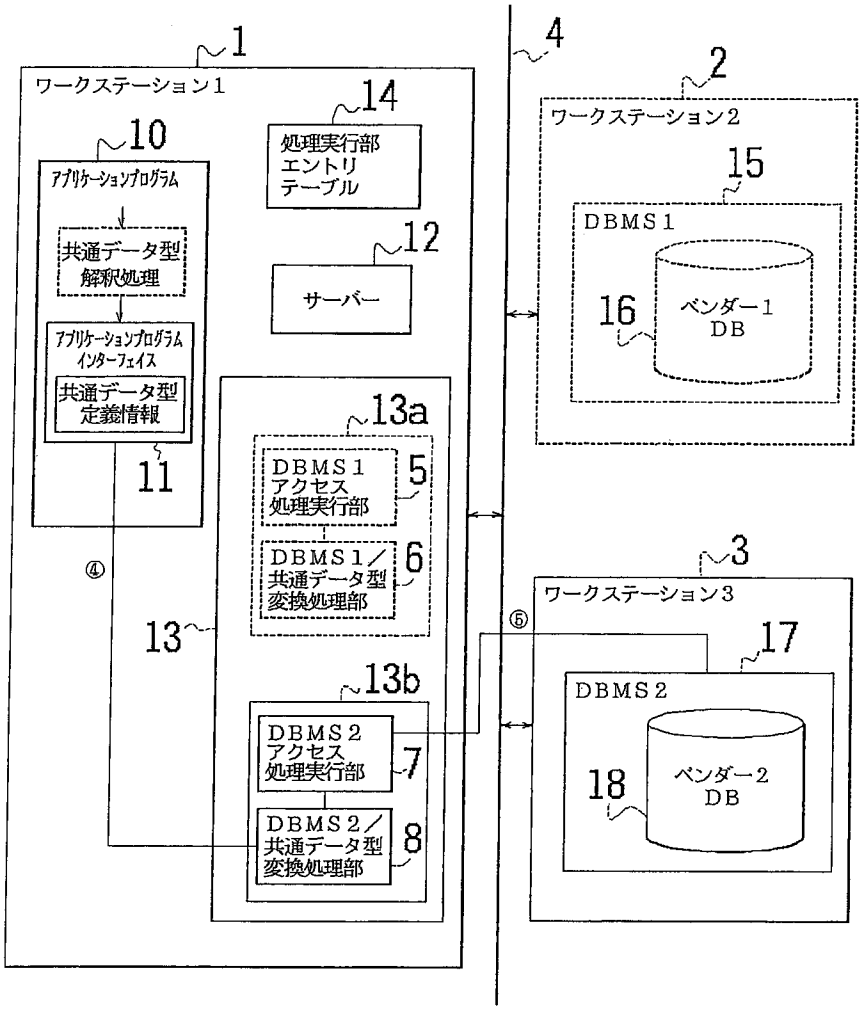
【図６】

図6

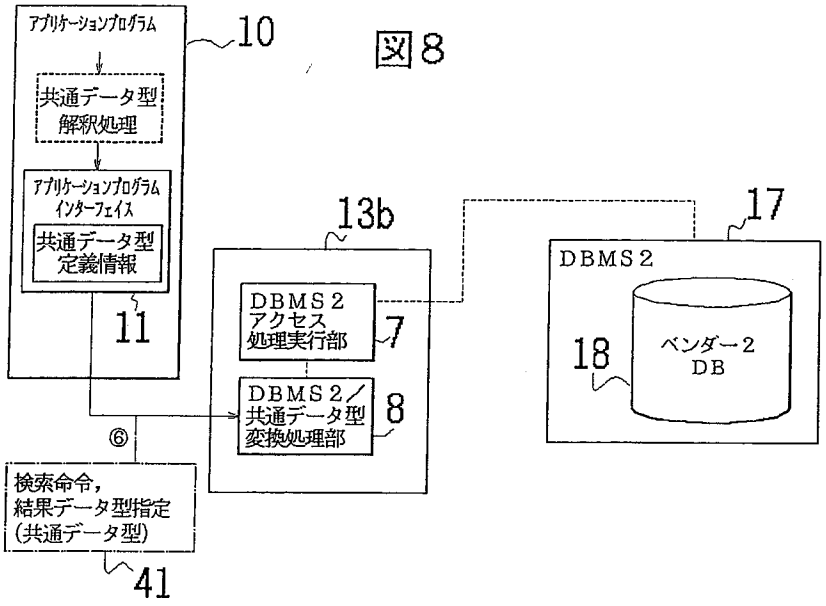

1
ワークステーション1
14
処理実行部
エントリ
テーブル
10
アプリケーションプログラム
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
12
サーバー
③
13a
DBMS1
アクセス
処理実行部
5
DBMS1／
共通データ型
変換処理部
6
④
13
13b
DBMS2
アクセス
処理実行部
7
DBMS2／
共通データ型
変換処理部
8
4
2
ワークステーション2
15
DBMS1
16
ベンダー1
DB
3
ワークステーション3
17
DBMS2
18
ベンダー2
DB

【図７】

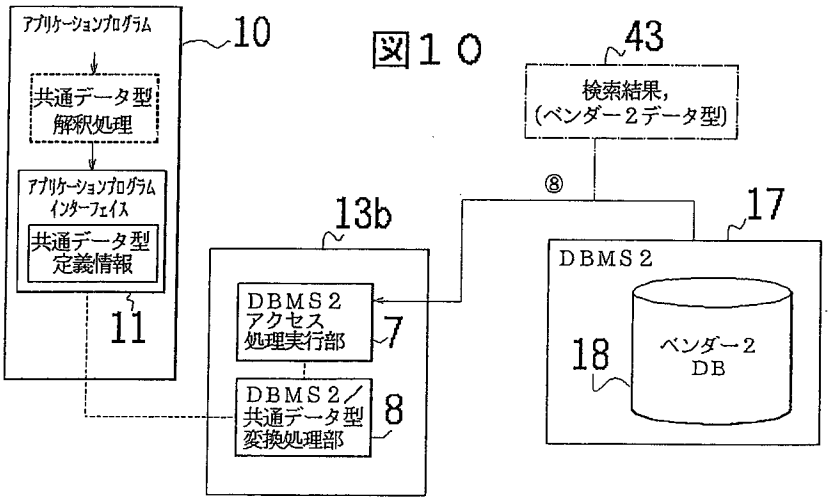
図７


1
ワークステーション１
10
アプリケーションプログラム
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
14
処理実行部
エントリ
テーブル
12
サーバー
13a
DBMS１
アクセス
処理実行部
5
DBMS１／
共通データ型
変換処理部
6
13
13b
DBMS２
アクセス
処理実行部
7
DBMS２／
共通データ型
変換処理部
8
④
4
2
ワークステーション２
15
DBMS１
16
ベンダー１
DB
3
ワークステーション３
⑤
17
DBMS２
18
ベンダー２
DB

【図８】


アプリケーションプログラム
10
図８
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
13b
DBMS２
アクセス
処理実行部
7
DBMS２／
共通データ型
変換処理部
8
17
DBMS２
18
ベンダー２
DB
⑥
検索命令，
結果データ型指定
(共通データ型)
41

【図１０】


アプリケーションプログラム
10
図１０
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
43
検索結果，
(ベンダー２データ型)
⑧
13b
DBMS２
アクセス
処理実行部
7
DBMS２／
共通データ型
変換処理部
8
17
DBMS２
18
ベンダー２
DB

【図１１】

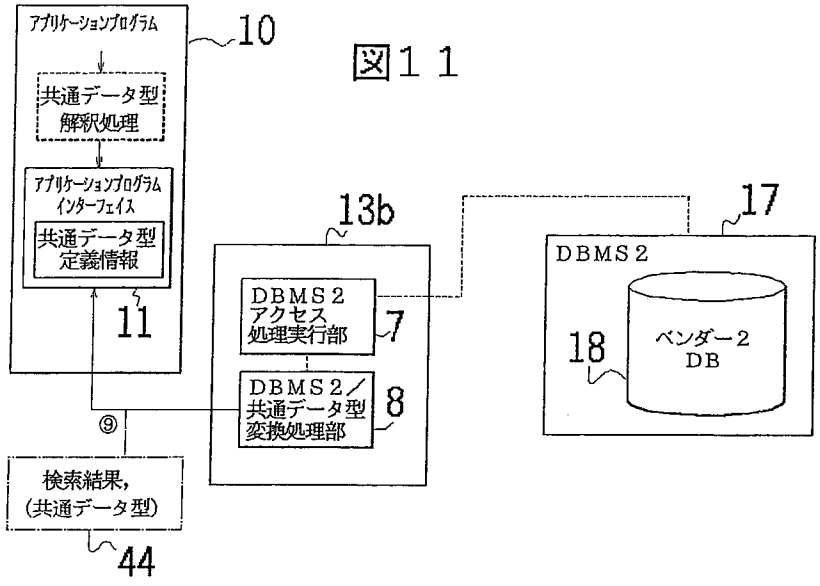


図１１

【図１２】

図１２

51 RCAデータ型	52 ORACLE	53 informix	54 Sybase
RCAChar	CHAR	CHAR	CHAR
RCALongchar	LONG	CHAR	TEXT
RCABin	ROW	CHAR	BINARY
RCALongbin	LONGRAW	CHAR	IMAGE
RCAInt	NUMBER	INTEGER	INT
RCALong	NUMBER	DECIMAL	MONEY
RCAFloat	NUMBER	FLOAT	FLOAT
RCANumber	NUMBER	CECIMAL	FLOAT
RCADate	DATE	DATE	DATETIME

【図１３】

図１３

ORACLE→RCAデータ型

ORACLE	RCAデータ型
CHAR	RCAChar
VARCHAR	RCAChar
LONG	RCALongchar
RAW	RCAChar
LONGRAW	RCALongbin
NUMBER	RCANumber
DATE	RCAdate
ROWID	RCARowid

【図１４】

図１４

ｉｎｆｏｒｍｉｘ→RCAデータ型

ｉｎｆｏｒｍｉｘ	RCAデータ型
CHAR	RCAChar
INTEGER	RCAInt
SMALLINT	RCAInt
FLOAT	RCAFloat
SMALLFLOAT	RCAFloat
DECIMAL	RCANumber
BCD	RCANumber
MONEY	RCANumber
DATE	RCADate
SERIAL	RCARowid

【図１５】

図１５

Ｓｙｂａｓｅ→RCAデータ型

Ｓｙｂａｓｅ	RCAデータ型
CHAR	RCAChar
VARCHAR	RCAChar
TEXT	RCALongchar
BINARY	RCABin
VARBINARY	RCABin
IMAGE	RCALongbin
INT	RCAInt
SMALLINT	RCAInt
TINYINT	RCAInt
BIT	RCAInt
MONEY	RCALong
FLOAT	RCAFloat
REAL	RCAFloat
DATETIME	RCADate
SMALLDATETIME	RCADate

H06-067867
Machine Translation from Japanese Patent & Utility Model Gazette DB
(http://www4.ipdl.inpit.go.jp/Tokujitu/tjsogodben.ipdl?N0000=115)

CLAIMS

[Claim(s)]
[Claim 1] A database access method of an application program which performs access processing of a database management system in an application program characterized by comprising the following.
An application program interface which performs interface processing in a form independent of classification of a database management system.
Each processing execution part which performs access processing to each to two or more database management systems of each.
A server which makes connection between a processing execution part and an application programming interface which perform access processing to a database management system of an operation target based on a connection request from said application programming interface.

[Claim 2] Based on a connection request from said application programming interface, A database access method of the application program according to claim 1 having further a table which stored management information to each database referred to when said server makes connection between a processing execution part and an application programming interface.

[Claim 3] In said application program interface. A database access method of the application program according to claim 1 having further a converter which performs conversion of a data type peculiar to each database management system, and a common data type to said each processing execution part including common data type defining information.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Industrial Application]This invention about the database access method of an application program in more detail, In the case where the application program accompanied by the access processing to the database management system (it is hereafter called DBMS for short) which operates a database is developed, The database access module in the application program can be developed efficiently, and, moreover, the access processing of the database in an application program is related with the database access method of the application program which can be performed well.

[0002]
[Description of the Prior Art]In order to develop conventionally an application program

including the processing which operates DBMS, With the gestalt of using the productivity tool for DBMS access provided from the vendor of each DBMS, the developer of an application program is developing software (application program). The productivity tool (it is hereafter called the productivity tool for DBMS for short) provided from the DBMS vendor which is this each donor of a database, DBMS of that each vendor of DBMS is accessed, and it is provided as the so-called access module of DBMS for the purpose of improvement using the database provided from the DBMS vendor concerned in the productivity of development of software.

[0003]However, to each productivity tool for DBMS. Since there is a difference peculiar to each, when the developer of an application program develops a program including the access processing of DBMS of two or more DBMS vendors, by DBMS made applicable to use. It must change to the productivity tool for DBMS of the correspondence, and the application program which used each productivity tool for DBMS individually must be developed.

[0004]Thus, when developing an application program including the processing which operates DBMS, by each DBMS. in order must be alike, respectively, to have to use for the productivity tool for DBMS of the correspondence and to have to perform program development -- the improvement in the productivity of software preparing work -- a figure -- there is no he. For this reason, in order to raise the productivity of program development, the application program development method which does not make it conscious of DBMS which is different in each is proposed. For example, in the "software program automatic generation method" indicated to JP,3-90933,A, the method which generates automatically the access language of the form suitable for the DBMS is proposed according to the attribution information of a file to access.

[0005]In order to often develop the software program which has the function to specify database processing according to the "software program automatic generation method" indicated to JP,3-90933,A, The classification of the database language generated from the attribution information of a file to access is distinguished, and the productivity tool which generates the database language of relevance automatically is provided. For this reason, the user can perform program development by using such a productivity tool, without being conscious of the structure of DBMS, the difference of classification, and a difference of the database language depending on this. This productivity tool is equipped with the target database table and the correspondence table of DBMS classification.
If a user directs the file name of the target database, corresponding to the database which a productivity tool accesses, a DBMS kind is distinguished from a database table and the access sentence by the database language of the form suitable for the DBMS is generated automatically.

[0006]
[Problem(s) to be Solved by the Invention]By the way, as mentioned above, in the productivity tool for DBMS in the former. In order for two or more different-species DBMS to develop the application program in the software operating environment which constitutes the distributed type database system connected on a network, First, the

program development after learning the directions of the productivity tool for every DBMS and being always conscious of each DBMS is needed. Therefore, program development accompanied by processing of the operation to a database cannot be performed efficiently, but there is a problem that the productivity of software development has not improved.

[0007]On the other hand, so that the example of a proposal given in above-mentioned JP,3-90933,A may see, the program development which the proposal of the productivity tool which can generate an application program without being conscious of DBMS which is different to each is made, was alike to some extent by performing program development using such a productivity tool, and raised the productivity of software is also possible.

[0008]However, when creating an application program including the access processing to several different DBMS in particular being unconscious of the feature of each DBMS, etc. using such a productivity tool, can expect the improvement in the productivity in program creation, but. If the application program which performs operation to two or more DBMS is created using this productivity tool, the created APUKESHON program, The program of the structure which has each access module to two or more DBMS in each by the application program side is created.

[0009]In this case, each program for access for DBMS created in the application program, Each is a program depending on each DBMS, and to perform each DBMS access from the unified user interface, it is necessary to link the user interface concerned and the generated program for each access for DBMS. Since it will have an access module for two or more DBMS by the application program side, If the number of DBMS of an operation target increases according to the kind of database, the performance of an application program will fall with DBMS of an accessing object, and the increase in a database table. Since there is no portion which absorbs a difference of the data type of each DBMS, there is a problem that a user interface does not serve as a program of the same contents.

[0010]When drawing 2 explains concretely, drawing 2 is a figure explaining the data base access processing by the application program created using a productivity tool for DBMS which was mentioned above. The workstation to which the application program accompanied by the access processing to a database runs 1 in drawing 2, It is a network with which 2 constitutes the 1st database donor's workstation, 3 constitutes the 2nd database donor's workstation, and 4 constitutes a Local Area Network. As for the database management system (DBMS1) of the vendor 1, and 16, the database management system (DBMS2) of the vendor 2 and 18 are the data base body parts of the vendor 2 the data base body part of the vendor 1, and 17 15. It is the access interfaces for the vendors 1 in which 20 accesses an application program body part and 21 accesses DBMS of the vendor 1, and 22 is access interfaces for the vendors 2 which access DBMS of the vendor 2.

[0011]Software operating environment is formed with such a system element, and the software operating environment of the application program 20 which operates here serves as distributed type database system to which two or more different-species DBMS is connected on a network. The application program 20 is generated by each above productivity tools for DBMS, and to the application program 20 here. The access interfaces 21 for the vendors 1 which access DBMS of the vendor 1 in order to operate the database of two different DB vendors, The access interfaces for the vendors 2 which access DBMS of the vendor 2 were being created by the program structure provided individually, respectively.

[0012]moreover -- in order to absorb a difference of a data type which is different by each DBMS -- an application -- in the KEYON program 20, it has a processing program which interprets the data type. therefore, an application -- the program of the vendor 1 data-type interpretation processing 20a is established to DBMS of the vendor 1, and the KEYON program 20 is a program of the structure where the program of the vendor 2 data-type interpretation processing 20b was established, to DBMS of the vendor 2. the processing program which performs these data type interpretations accepts necessity -- an application -- it is embedded in the KEYON program 20.

[0013]When each DBMS is accessed with the access interfaces for each vendors, The data type interpretation treating part for each vendors (20a, 20b) is linked with each access interfaces, The data type in an application program is interpreted, the data type of an application program is set by the data type for each vendors, and access interfaces (21, 22) perform access processing to each DBMS.

[0014]Therefore, as a result of providing the application program 20 here with the access interfaces for each vendors of an operation target, and the processing program of the data type interpretation processing for each vendors, the program itself becomes heavy and overall performance falls. The program of the data type interpretation processing equipped corresponding to the access interfaces of DBMS for each of these vendors when the target database to access increases, It must create by the developer side of an application program, and there is a problem that that it cannot respond to future upgrade etc. easily etc. has the bad conservativeness of a program.

[0015]Were made in order to solve the above problems of this invention, and the purpose of this invention, In the case where the application program accompanied by the access processing to DBMS which operates a database is developed, The database access module in the application program can be developed efficiently, And the access processing of the database of each application program can carry out well as the whole system, and it is in providing the database access method of the application program whose conservativeness of the program improved.

[0016]Other purposes of this invention have access from the same application program in providing the database access method of the application program which becomes possible with common access interfaces to two or more DBMS.

[0017]When other another purposes of this invention provide the productivity tool (common library) whose access from the same application program is attained to two or more DBMS, The developer of an application program is not made conscious of the difference in the structure of each DBMA, or each application interface, It is in providing the database access method of the application program of the structure (the difference in a data type will be absorbed) where it is not made conscious of the difference in a data type.

[0018]
[Means for Solving the Problem]This invention in order to attain the above purposes a database access method of an application program of this invention, A database access method of an application program which performs access processing of a database management system (15, 17) in an application program (10) is characterized by comprising:
An application program interface (11) which performs interface processing in a form independent of classification of a database management system.
Each processing execution part (13) which performs access processing to each to two or more database management systems of each.
A server (12) which makes connection between a processing execution part and an application programming interface which perform access processing to a database management system of an operation target based on a connection request from said application programming interface.

[0019]Said server (12) here based on a connection request from said application programming interface, When making connection between a processing execution part and an application programming interface, it has further a table (14) which stored management information to each database to refer to.

[0020]Said application program interface (11) was further equipped with a converter (6, 8) which performs conversion of a data type peculiar to a database management system, and a common data type to said each processing execution part (13) including common data type defining information.

[0021]
[Function]According to the database access method of the application program of this invention. In the application program (10) which performs access processing of two or more database management systems (15, 16). In the form independent of the classification of a database management system, the application program interface (11) which performs interface processing with each processing execution part is established, It has each processing execution part (13) which performs access processing between each database management systems which each application program can use in common in the system which provides the software operating environment by which the application program (10) operates, and a server (12).

[0022]The application program interface (11) with which the application program (10) was equipped, Interface processing independent of the classification of a database

management system is performed, and each processing execution part (13) performs each access processing (5, 7) to two or more database management systems of each. A server (12) based on the connection request from an application programming interface (11), Connection between the individual processing execution part (13a, 13b) which performs access processing to the target database management system, and the application programming interface (11) which advanced the connection request is made.

[0023]For this reason, it has the table (14) which stored the management information corresponding to each database referred to when a server makes connection between a processing execution part and an application program, and it is, With reference to the contents of the table (14), as for a server (12), a server makes connection between a processing execution part and an application programming interface.

[0024]Although an application program interface (11) performs interface processing for which it does not depend on the classification to two or more database management systems in an application program (10), Under the present circumstances, to an operator, the restrictions to a difference of a data type are absorbed and an application program interface (11) performs common data type interface processing so that processing of the database operation by common operation may be possible. For this reason, common data type defining information is included in the application program interface (11), and each processing execution part (14) is further equipped with the converter (6, 8) which performs data type conversion using this defining information. A converter (6, 8) performs conversion of a data type peculiar to a database management system, and a common data type. The common data type by this defining information performs connection between an application program interface (11) and each processing execution part (14), Then, data type conversion is performed and each processing execution part (14) connects with the peculiar data type of each database management system between each database management system.

[0025]Therefore, since the difference is absorbed by the converter (6, 8), the data type between each database management systems the developer of an application program, Especially, being unconscious of the difference in the structure of each DBMA, or each application program interface, and also the difference in a data type, an application program can be developed, the development burden is eased, and the productivity of development can be raised.

[0026]Thus, when data base access processing is performed in an application program, A server based on the connection request from an application programming interface, Connection between the processing execution parts which perform access processing to target DBMS is made, and access processing to one data **-SU of one DBMS vendor which is applicable is performed via an application program interface and a processing execution part. When the target databases to access differ, a server makes connection with the processing execution part corresponding to the DBMS, and access processing to a database is performed via a common application program interface from an application program.

[0027]Here, the application program side is equipped only with the application program interface.
Access processing to the database followed and kicked to an application program is performed using an application program interface, each processing execution part, and a server.
For this reason, when developing an application program including the operation to a database, especially to an application program. Since what is necessary is to have only the application program interface common to operation of each DBMS, he needs to be conscious of neither the structure of each DBMS, nor the application program interface for each DBMS to an application program developer.

[0028]Thereby, since the application program developer needs to be conscious of neither the structure of each DBMS, nor the application program interface for each DBMS and he does not need to be conscious of a difference of a data type, the burden of the developer of an application program is eased. For this reason, the productivity of software development can be raised. Since an application program interface should publish only the connection request to the database to a server also when an application program performs operation to two or more databases, The application program itself becomes heavy and performance does not fall. When the target database to access newly increases, Create only the processing execution part for DBMS to DBMS which newly increased using the productivity tool for which it is provided from a DBMS vendor, and the processing execution part which contained the data type converter when required is created, What is necessary is just to install in a system, and it can respond, without performing correcting work of an application program exceptionally. For this reason, it can respond to future upgrade etc. easily and the conservativeness of a program is good.

[0029]
[Example]Hereafter, one example of this invention is concretely described with reference to drawings. Drawing 1 is the Plock figure showing the composition of the important section of the database access method of the application program concerning one example of this invention. The workstation to which the application program accompanied by the access processing of a database runs 1 in drawing 1, It is a network with which 2 constitutes the 1st database donor's workstation, 3 constitutes the 2nd database donor's workstation, and 4 constitutes a Local Area Network. An application program interface and 12 show a server, 13 (13a, 13b) shows each processing execution part for DBMS vendors, and, as for an application program and 11, 10 shows the processing execution part entry table 14, respectively. As for the database management system (DBMS) of the vendor 1, and 16, the database management system (DBMS) of the vendor 2 and 18 are the data base body parts of the vendor 2 the data base body part of the vendor 1, and 17 15. Each processing execution part 13 for DBMS vendors corresponds to each access processing for vendors (13a, 13b), It has the DBMS1 access-processing execution part 5, the DBMS1/common data type conversion treating part 6, the DBMS2 access-processing execution part 7, and the DBMS2/common data type conversion treating part 8, respectively.

[0030]The software operating environment which the application program 10 here runs serves as a network system by which each work souchong 1, 2, and 3 was mutually connected with the network 4, as shown in drawing 1.
Two or more different-species DBMS constitutes the distributed type database system connected on the network from the workstations 2 and 3.
also when operating the database in DBMS of two different DB vendors, the application program 10 does not need the access interfaces which were alike, respectively and corresponded, but is used in common -- it is accepted application program interface 11 and is easy to come out. In database processing, in order for a common data type to perform database operation, common data type interpretation processing is performed and a link with the application program interface 11 is performed. The workstation 1 to which the application program 10 accompanied by the access processing of a database runs is equipped with the server 12, each processing execution part 13 for DBMS vendors, and the processing execution part entry table 14 so that it may illustrate.

[0031]As a processing execution part for performing access processing to each DBMS in the processing execution part 13 for DBMS vendors, The data type conversion between the 1st processing execution part 5 for DBMS vendors, the DBMS1/common data type conversion treating part 6 which performs data type conversion between the 1st data type for DBMS, and a common data type, the 2nd processing execution part 7 for DBMS vendors, the 2nd data type for DBMS, and a common data type. It has the DBMS2/common data type conversion treating part 8 to perform. Here common data type defining information is included in the application program interface 11.
It is referred to if needed.

[0032]To the application program 10 accompanied by the access processing to a database. Program development in the gestalt to which only the application program interface 11 was added for access of a database is performed, The application program 10 concerned will run on the workstation 1 by which it had the server 12, each processing execution part 13 (5, 6, 7, 8) for DBMS vendors, and the processing execution part entry table 14.

[0033]The application programming interface 11 is a program module which provides the interface function non-depending for DBMS in application program development.
The program development person can perform program development, without being conscious of the structure of each DBMS, each application program interface for DBMS, and a data type, also when treating two or more DBMS by becoming skilled about this common interface function.

[0034]The server 12 watches the connection request from the application program interface 11 in the workstation 1 concerned, When the connection request is published, the function to perform a connection service with each access processing part (13a, 13b) in each processing execution part 13 for DBMS vendors which is applicable is provided. Here, in order to absorb a difference of a data type, connection between the application program interface 11 and each DBMS access processing execution part (5, 7) is made via

each DBMS / common data type conversion treating part (6, 8).

[0035]Each DBMS access processing execution part of the processing execution part 13 for DMBS vendors is created by the productivity tool from each vendor using each application interface for vendor DBMS.
In the workstation 1 concerned, the difference of the access method of each DBMS and a difference of a data type are absorbed, and the access processing (issue, receipt) function to DBMS of the object which operates a database is provided.

[0036]The processing execution part entry table 14 is a table where the management information referred to when the server 12 performs connection processing to the database connection request from the application program 10 is registered.
To the DBMS name by which a connection request is carried out, the processing execution part name for DBMS vendors is made to correspond, and it has registered. Correspondence of whether to make connection with the access processing part (13a, 13b) of which DBMS of the processing execution part 13 for DBMS vendors is referred to to the connection request out of the processing in the application program 10.

[0037]Drawing 3 is a figure showing the example of composition of a processing execution part entry table. If the processing execution part entry table 14 is explained concretely, as shown in drawing 3, it comprises the DBMS name field 31 which registers the name of DBMS, and the DBMS access processing execution part name field 32 which registers the access tool of correspondence DBMS.
It is the entry table which has registered the name of the ****** BE ** DBMS access processing execution part corresponding to the DBMS name.
It is the management information which has managed each connectable DBMS in this distributed database system. As mentioned above, when the connection request which specified the DBMS name from the application program interface 11 is published, the server 12 refers to the processing execution part entry table 14, A DBMS access processing execution part program name corresponding from a DBMS name is got to know, and the DBMS access processing execution part program of relevance is started.

[0038]Next, the outline of access processing over the database in the application program 10 which operates in such operating environment is explained. First, if processing advances to the step which common data type interpretation processing in database operation is performed, and performs access processing of DBMS in processing of the application program 10, The application program interface 11 publishes a connection request to DBMS to the server 12 by interface processing with an application program body part. The server 12 refers to the matching information in the processing execution part entry table 14 based on the connection request from the application programming interface 11, The access processing part of correspondence of the processing execution part 13 for DBMS Venter which performs access processing to the target database management system is started, and then connection between the body parts of the application program 10 is made via the application program interface 11. Thereby, the DBMS access processing part 13b of the connected processing execution part 13 for

DBMS Venter performs access processing to DBMS of an operation target.

[0039]Thus, when database access is performed in the application program 10, The server 12 based on the connection request from the application programming interface 11, The processing execution part for individual DBMS vendors (13a or 13b) which performs access processing to target DBMS is started, For example, connection between the access processing execution part 13b for DBMS vendors and the application programming interface 11 is made, and the processing execution part 13 for DBMS vendors performs access processing between target DBMS, and makes connection in the meantime. Therefore, in data base access processing, data **-SU operation of target DBMS will be performed via the application program interface 11 and the processing execution part 13 for DBMS vendors.

[0040]Next, with reference to drawing 4 - drawing 7, connection between each treatment modules is made and each state which seems to follow access processing to a database one by one is explained. Drawing 4 is a figure showing the state at the time of being immediately after executing an application program.
Drawing 5 is a figure showing the state at the time of an application program performing the connection request of an application programming interface.
Drawing 6 is a figure showing the state at the time of the server 1 starting the processing execution part for DBMS vendors of an operation target.
Drawing 7 is a figure showing a state when an application program performs an access request through an application programming interface.

[0041]First, when the application program 10 performs access operation to the database provided from each vendor from an initial state as shown in drawing 4, as shown in drawing 5, First, the application program 10 publishes connection-request ** to the server 12 via the application programming interface 11. If connection-request ** is published, the server 12 will catch this connection-request **. The server 12 distinguishes the demand parameter (with which DBMS does it connect?) of connection-request **, and performs processing ** which becomes final and conclusive the processing execution part program name for DBMS vendors which performs access processing from the contents to target DBMS with reference to the processing execution part entry table 14.

[0042]starting ** Next, it will be in the state where it is shown in drawing 6, and the server 12 will carry out the DBMS2 access-processing execution part 7 concerned to carry out from the program name fixed with the processing execution part entry table 14. The started DBMS2 access-processing execution part 7, In order to make connection between the application program interfaces 11 of the application program 10, The DBMS2/common data type conversion treating part 8 which performs data type conversion between the data type of DBMS2 of correspondence and a common data type is started, and path ** which performs data base access processing in the meantime is established via this DBMS2/common data type conversion treating part 8. At this time, the server 12 will be in the state where other application programs wait for issue of the following connection request which requires through that application program interface.

[0043]And the DBMS2 access-processing execution part 7 which the following state changed into the state where it is shown in drawing 7, and was started, Path ** which performs connection processing with DBMS2 (17) made into an object by the demand from the application program interface 11, and performs data base access processing in the meantime is established. As a result, it lets access path [with the application program interface 11 established previously] **, and access path ** of DBMS2 (17) established after that pass, The application program 10 via each treatment module (the DBMS2 access-processing execution part 7, the DBMS2/common data type conversion treating part 8) of the application program interface 11 and the processing execution part 13b for DBMS2 vendors, It will be in the state of performing access processing to DBMS2 (17) of the data base body part 18 of an operation target. After the access processing to DBMS2 (17) here is completed, the application program 10 lets the application program interface 11 pass after that, After performing a disconnect request, the DBMS2 access-processing execution part 7 performs on-off processing, processing is ended, and it returns to the state where it is shown in drawing 4.

[0044]Next, with reference to drawing 8 - drawing 11, order is explained for how the difference in a data type peculiar to DBMS of each vendor is absorbed in the access processing of the database from an application program later on like the above-mentioned explanation. Drawing 8 is a figure showing a state when a retrieving instruction is published from an application program.
Drawing 9 is a ** figure about a state when publishing a retrieving instruction from a DBMS access processing execution part to DBMS of an operation target.
Drawing 10 is a figure showing a state when the search results from DBMS of an operation target have returned to the DBMS access processing execution part which published the retrieving instruction.
Drawing 11 is a figure showing a state in case a DBMS access processing execution part returns search results to the application program which published the retrieving instruction.

[0045]Here, the case where a retrieving instruction is published from an application program to a database is made into an example, and it explains concretely. First, as shown in drawing 8, the retrieving instruction 41 of the database published in the application program 10 is published from the application program interface 11, but. In this case, it specifies receiving search results with a common data type, and the retrieving instruction 41 is published to the DBMS2 access processing part 13b which performs access processing of DBMS2. Namely, in application plog RARAMU 10. it specifies that the data type of search results which he receives in the application program interface 11 is obtained from the common data type defined by defining information, for example with a common string type, and it says it with the parameter of the retrieving instruction 41, and a retrieving instruction is published.

[0046]The DBMS2 access-processing execution part 7 and the DBMS2/common data type conversion treating part 8 to which the DBMS2 access processing part 13b performs access executive operation to DBMS2 of an accessing object by a previous connection request are started.

The DBMS2 access-processing execution part 7 changes and receives the retrieving instruction 41 depended on this common data type from a common data type to the retrieving instruction of the data type for the vendors 2 via the DBMS2/common data type conversion treating part 8.

[0047]Next, the DBMS2 access-processing execution part 7 changed and received to the retrieving instruction of the data type for the vendors 2, Access processing to the database management system (DBMS2) 17 is performed, and as shown in drawing 9, the retrieving instruction 42 of the data type for the vendors 2 is transmitted to the database management system (DBMS2) 17. Thereby, in the database management system (DBMS2) 17, line practice and its search results are returned for database retrieval processing to the vendor 2 data-base-body part 18 by the retrieving instruction 42.

[0048]As a result, as shown in drawing 10, the search results 43 are returned from the database management system (DBMS2) 17 with the data type for the vendors 2 to search results. The search results 43 in the data type for the vendors 2 are received in the DBMS2 access-processing execution part 7. Next, the DBMS2 access-processing execution part 7 performs data type conversion to common data type search results for the search results 43 in the data type for the vendors 2 via DBMS2 / common data type conversion treating part 8, as shown in drawing 11, As the common data type search results 44, search results are returned to the application program interface 11. In the application program 10, this uses common data type search results. Thus, conversion of each data type in the retrieval processing of a database is performed. When performing alter operation to a database and performing update operation, it is processed by performing conversion to the data type corresponding to DBMS of each vendor from a common data type.

[0049]drawing 12 -- an application -- it is a figure showing the example of composition of the common data type defining information provided in a KESHO program interface. This defining information is defined as drawing 12 as a correspondence table of the defining information of common data type 51, and this common data type 51 and the corresponding data type of DBMS of each vendor of each so that it may be shown. The RCA data type which is common data type 51 here, The data type 52 of the 1st "ORACLE" of DBMS1 access processing is possible for, the data type 53 of the 2nd "informix" of DBMS2, and the data type 54 of the 3rd "Sybase" of DBMS3 are matched by this system. Conversion of these data types is not matched with 1 to 1, and is arbitrarily matched according to the contents of the APUKESHON program.

[0050]Drawing 13, drawing 14, and drawing 15 are the figures showing the example correspondence-related in case the peculiar data type of each DBMS is changed into the common data type of an application program interface in retrieval processing. The conversion relation from the data type of a database "ORACLE" to a common data type is shown in drawing 13, and the conversion relation from the data type of a database "informix" to a common data type is shown in drawing 14. The conversion relation from the data type of a database "Sybase" to a common data type is shown in drawing 15. A complicated data type peculiar to each database can be easily specified by specification

with a common data type by using the database access method of the application program of this example so that clearly from the conversion relation of these figures.

[0051]As explained above, when the application program 10 performs access processing accompanied by operation of a database, Based on the connection request from the application programming interface 11, The server 12 makes connection between the processing execution parts 13 for DBMS vendors which perform access processing to target DBMS, Access processing to the one data **-SU body part 18 used as an operation target is performed via each treatment module of the application program interface 11 and the processing execution part 13 for DBMS vendors. Under the present circumstances, it is alike and a difference of a data type is also absorbed in each processing execution part 13 for DBMS vendors.

[0052]When the target databases accessed with each application program 10 differ, The server 12 makes connection with the processing execution part 13 for DBMS vendors corresponding to the DBMS, and the access processing to a database carries out via the application program interface 11 of common specifications from an application program. For this reason, have with the server 12 and the connection request from an application program interface is received, The processing execution part entry table 14 which gives information for a server to make connection between the processing execution part 13 for DBMS vendors and the application program interface 11 is formed.

[0053]It is as follows when the effect by this example described above is summarized. (1) In the former, since the application program interfaces of each DBMS for vendors differed, the application program also had to become a thing depending on it, and had to develop the application program individually, but. According to an access method like this example, the application program which accesses two or more DBMS is the same, and ends. Namely, with an application program interface, when an application program publishes the connection request to DBMS, it should just specify DBMS. Although the data type for each DBMS needed to be interpreted by processing of the application program, respectively, it becomes possible by using a common data type to treat the data type for each DBMS only by performing a common data type interpretation.

[0054](2) In the former, if application program interface specification in each Kaihatsu ** tool for vendor DBMS was not learned, development of each application program for DBMS was impossible, but. By using the access method by this example, an application program developer only learns the utilizing method of the application program interface of common specifications, and the development of the application for DBMS of each vendor of him is attained. For this reason, software development efficiency improves. It will be provided with the access interfaces which can access DBMS of the vendor 1, DBMS of the vendor 2, and which if in other words one application program is created.

[0055](3) When new DBMS correspondence will be upgraded to an application program in the future, as a processing execution part for vendor DBMS of the correspondence, What is necessary is just to add a DBMS access processing execution part and the data type conversion process part between the DBMS data type for vendors concerned, and a

common data type, and it is not necessary to modify the application program itself. If the new DBMS name of a vendor and a DBMS access processing execution part name are added to a processing execution part entry table when the new DBMS processing execution part for vendors is created, access to new DBMS will be attained.

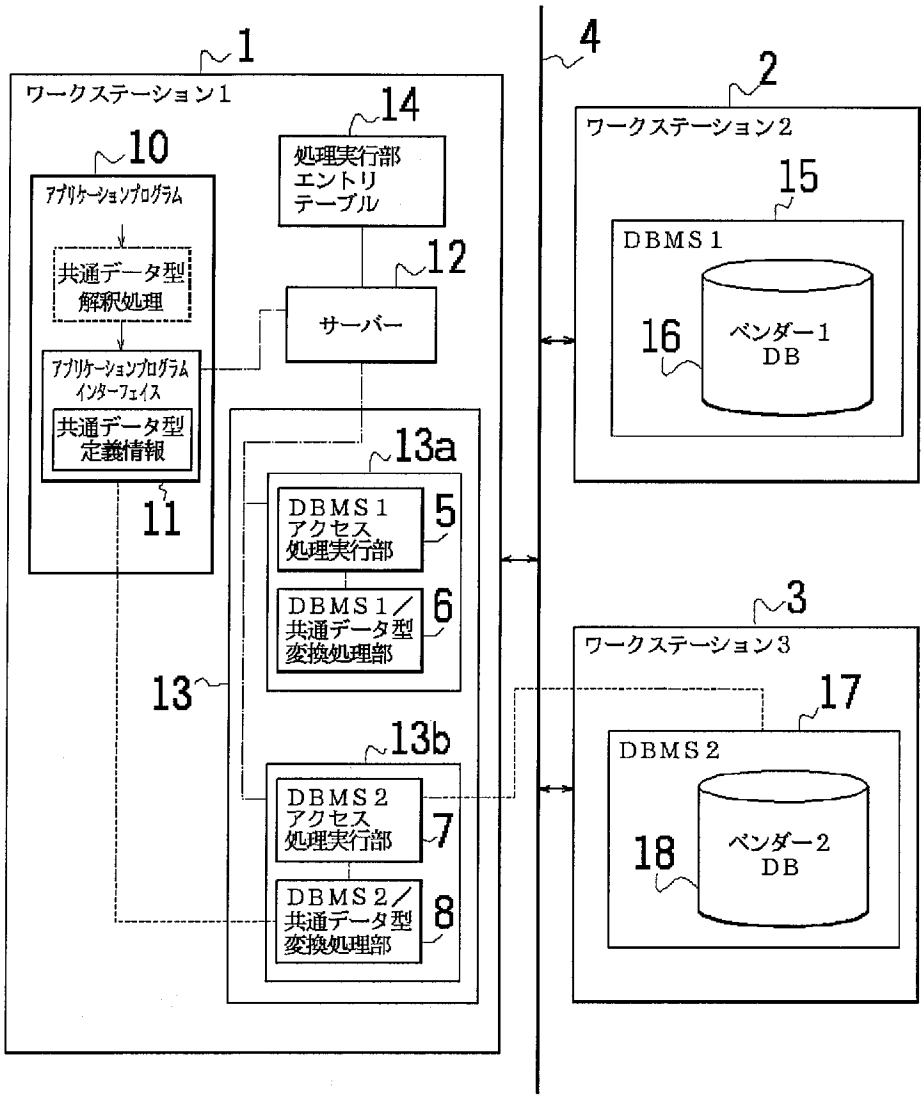
[0056]
[Effect of the Invention]As mentioned above, as explained, according to the database access method of the application program of this invention, to each application program side. Have only an application program interface fundamentally, and since ** is good, When developing an application program including the operation to a database, he needs to be conscious of neither the structure of each DBMS, nor each application program interface for DBMS to an application program developer. A common data type data type conversion process part is provided, and he does not need to be conscious of a difference of a data type by **.

[0057]Therefore, the application program developer can develop especially a program including the access processing of a database being unconscious of the structure of each DBMS, or each application program interface for DBMS. The burden is eased and the application program developer can raise the productivity of development. Since an application program interface should publish only the connection request to the database to a server when an application program performs operation to two or more databases, the performance of an application program does not fall.

DRAWINGS

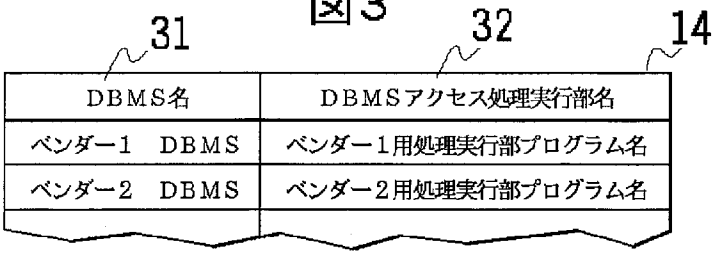
[Drawing 1]

図1



[Drawing 3]

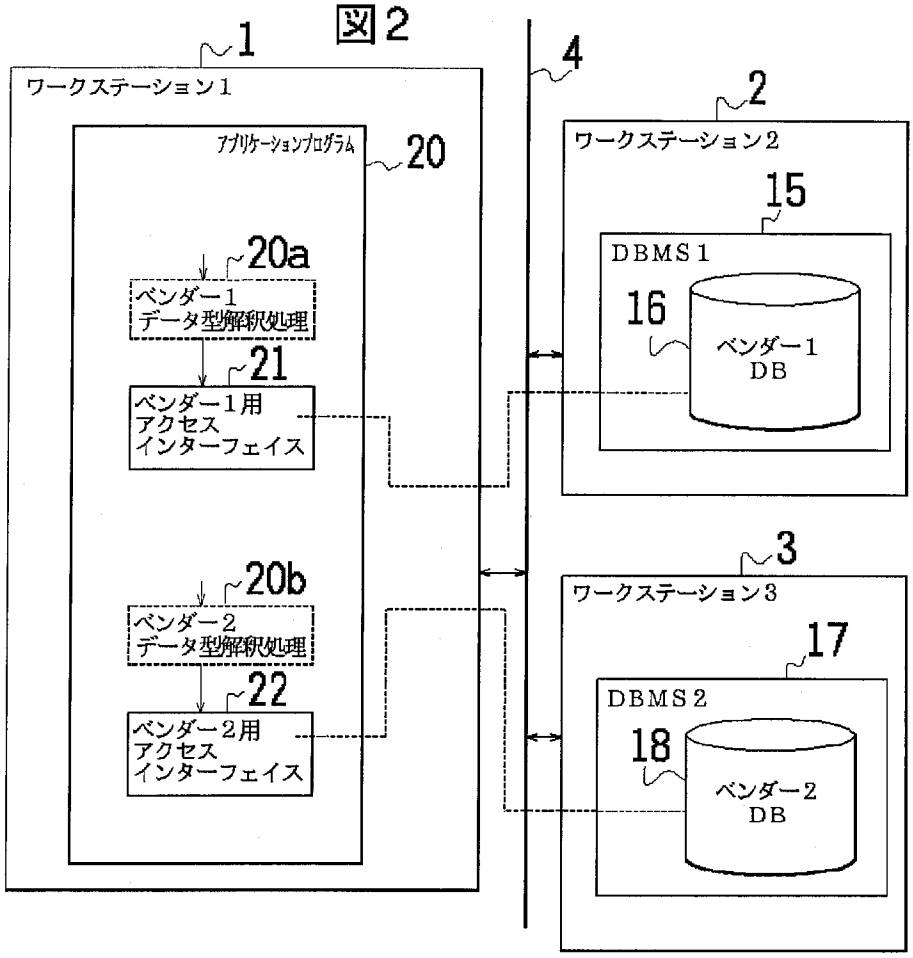
図3



DBMS名	DBMSアクセス処理実行部名
ベンダー1　DBMS	ベンダー1用処理実行部プログラム名
ベンダー2　DBMS	ベンダー2用処理実行部プログラム名

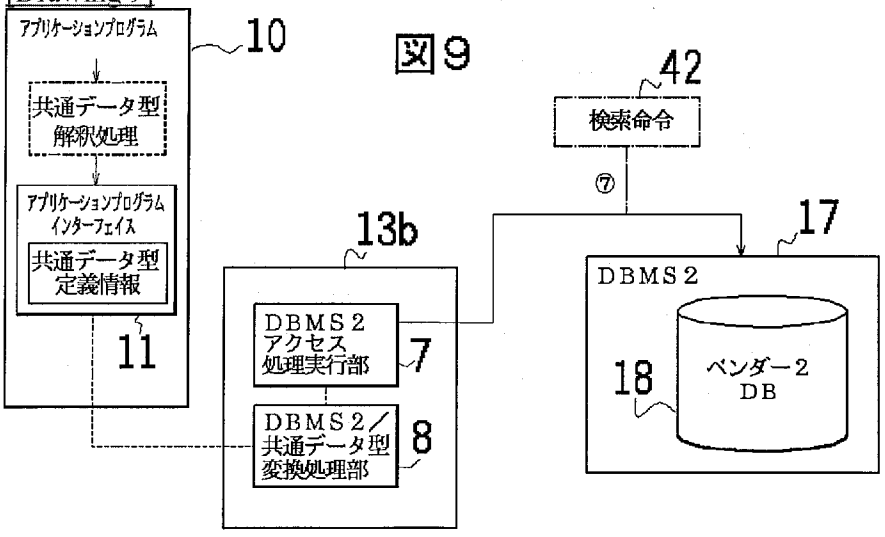
[Drawing 2]

図2



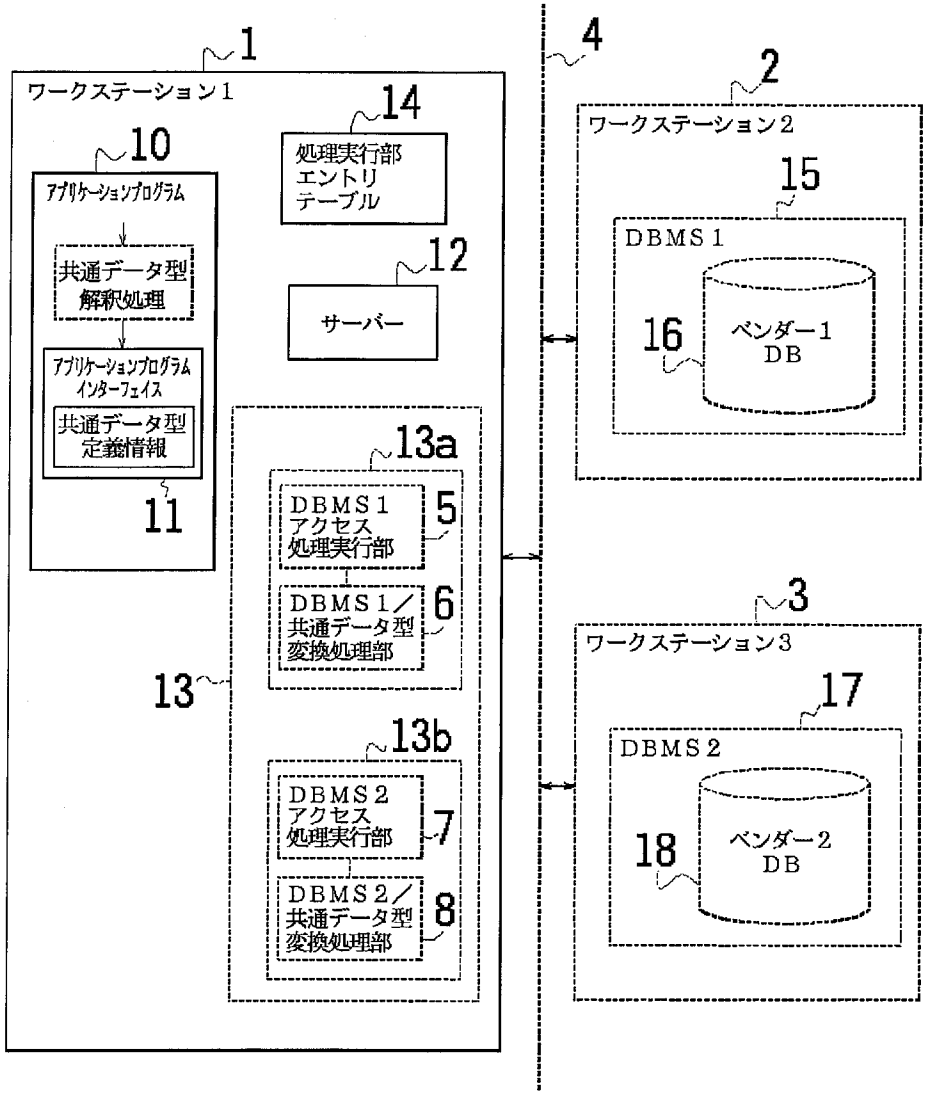
[Drawing 9]

図9



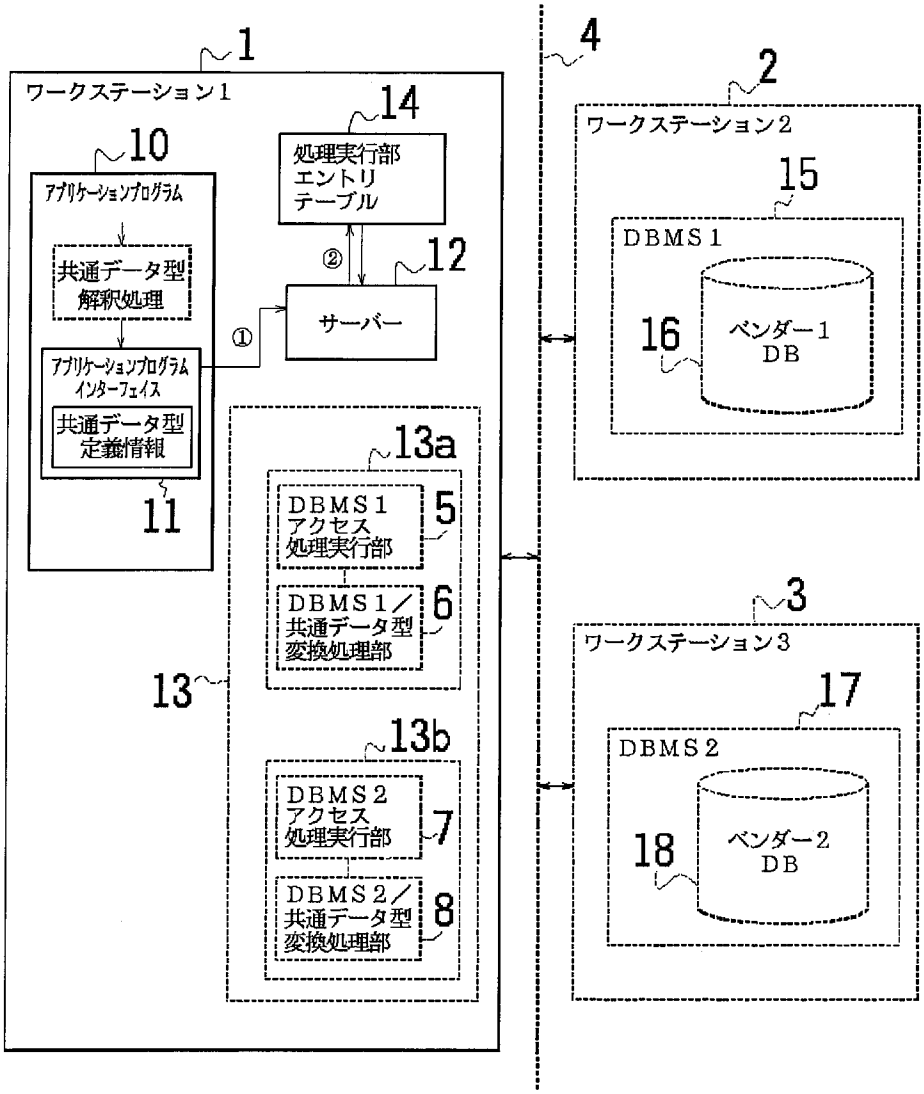
[Drawing 4]

図4


1
ワークステーション1
14
処理実行部
エントリ
テーブル
10
アプリケーションプログラム
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
12
サーバー
13a
DBMS1
アクセス
処理実行部
5
DBMS1／
共通データ型
変換処理部
6
13
13b
DBMS2
アクセス
処理実行部
7
DBMS2／
共通データ型
変換処理部
8
4
2
ワークステーション2
15
DBMS1
16
ベンダー1
DB
3
ワークステーション3
17
DBMS2
18
ベンダー2
DB

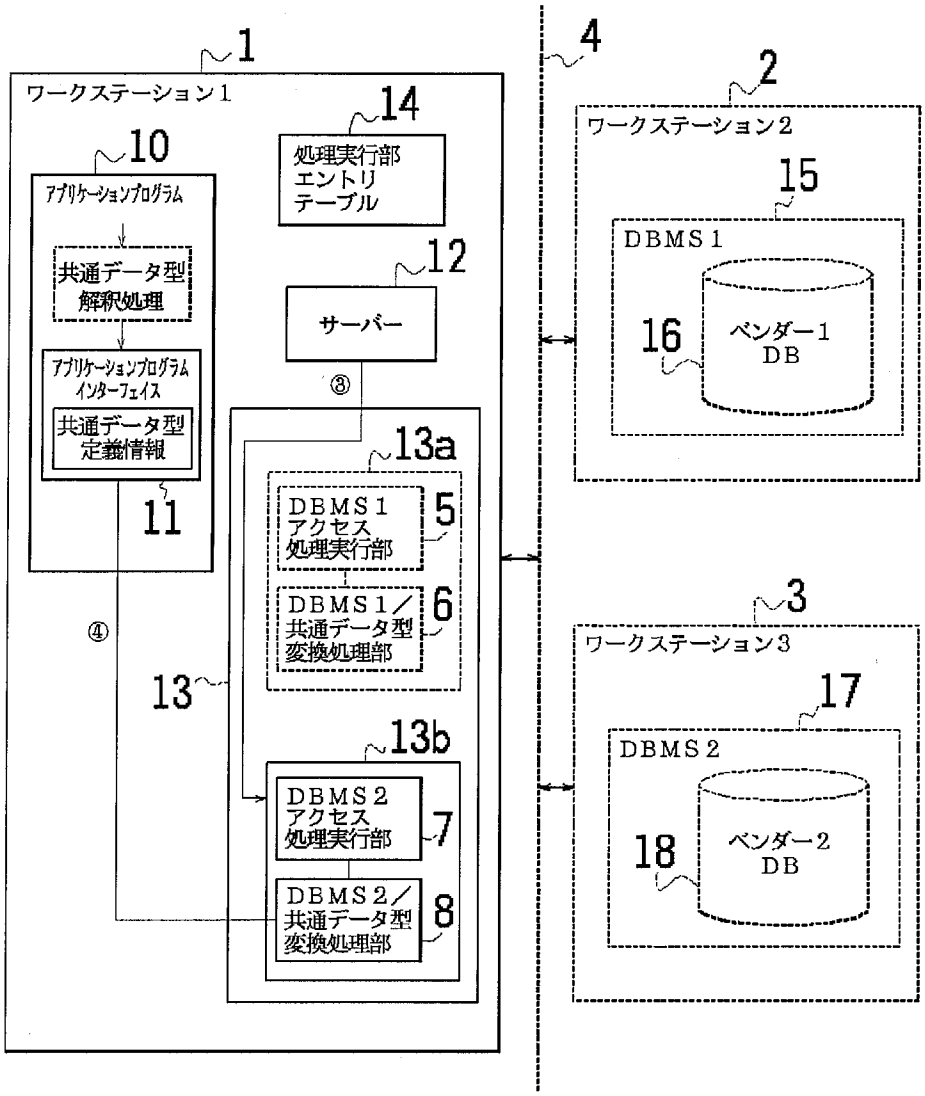
[Drawing 5]

図5


1
ワークステーション1
14
処理実行部
エントリ
テーブル
10
アプリケーションプログラム
共通データ型
解釈処理
②
12
サーバー
①
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
13a
DBMS1
アクセス
処理実行部
5
DBMS1／
共通データ型
変換処理部
6
13
13b
DBMS2
アクセス
処理実行部
7
DBMS2／
共通データ型
変換処理部
8
4
2
ワークステーション2
15
DBMS1
16
ベンダー1
DB
3
ワークステーション3
17
DBMS2
18
ベンダー2
DB

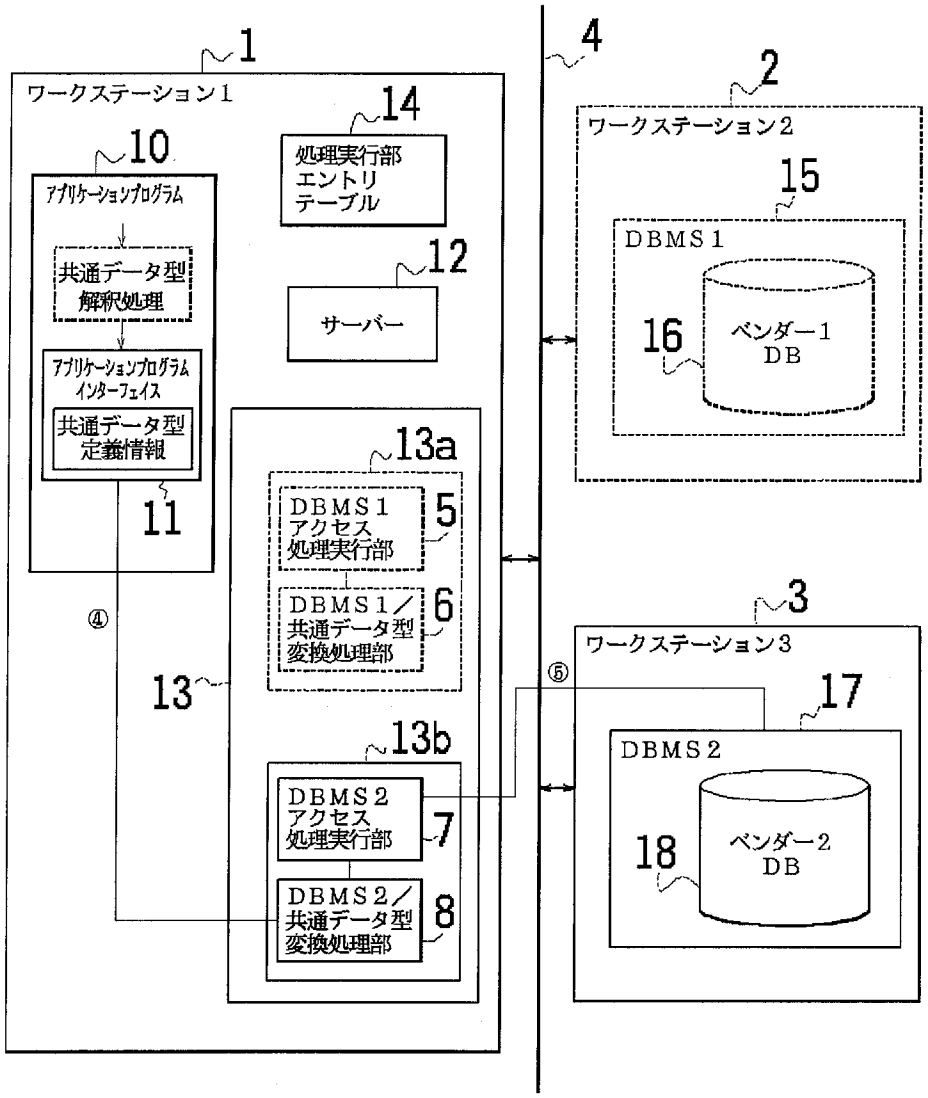
[Drawing 6]

図6


1
ワークステーション1
14
処理実行部
エントリ
テーブル
10
アプリケーションプログラム
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
12
サーバー
③
13a
DBMS1
アクセス
処理実行部
5
DBMS1／
共通データ型
変換処理部
6
④
13
13b
DBMS2
アクセス
処理実行部
7
DBMS2／
共通データ型
変換処理部
8
4
2
ワークステーション2
15
DBMS1
16
ベンダー1
DB
3
ワークステーション3
17
DBMS2
18
ベンダー2
DB

[Drawing 7]

図7


1
ワークステーション1
14
処理実行部
エントリ
テーブル
10
アプリケーションプログラム
共通データ型
解釈処理
アプリケーションプログラム
インターフェイス
共通データ型
定義情報
11
12
サーバー
13a
DBMS1
アクセス
処理実行部
5
DBMS1／
共通データ型
変換処理部
6
④
13
13b
DBMS2
アクセス
処理実行部
7
DBMS2／
共通データ型
変換処理部
8
4
2
ワークステーション2
15
DBMS1
16
ベンダー1
DB
3
ワークステーション3
⑤
17
DBMS2
18
ベンダー2
DB

[Drawing 8]

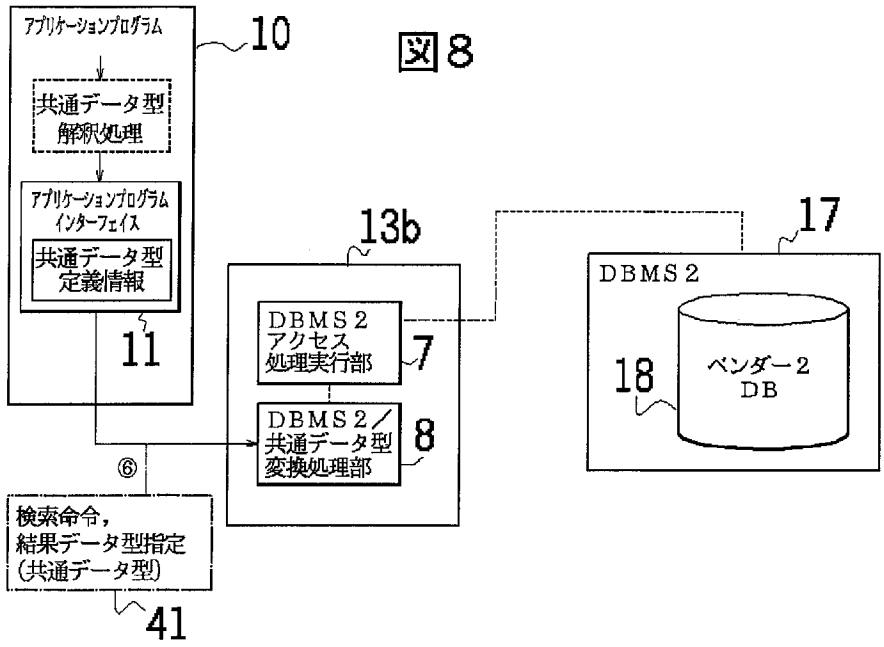


図8

[Drawing 10]

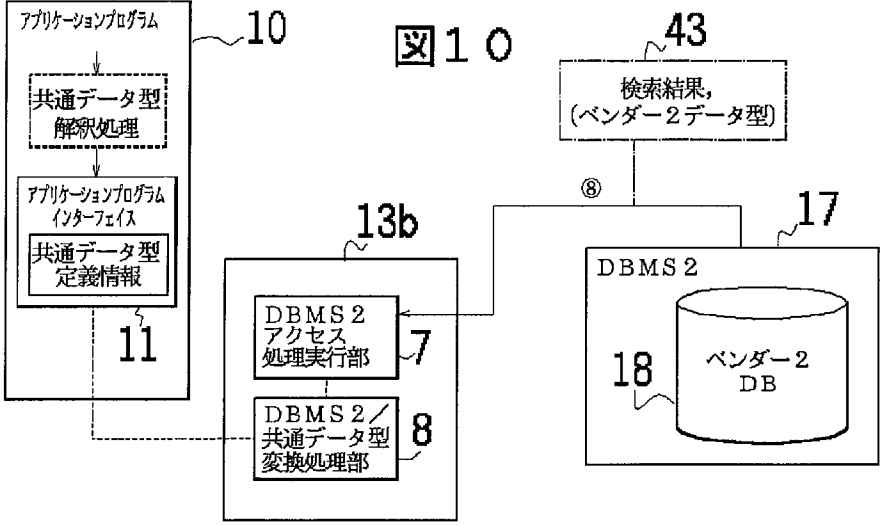


図10

[Drawing 11]

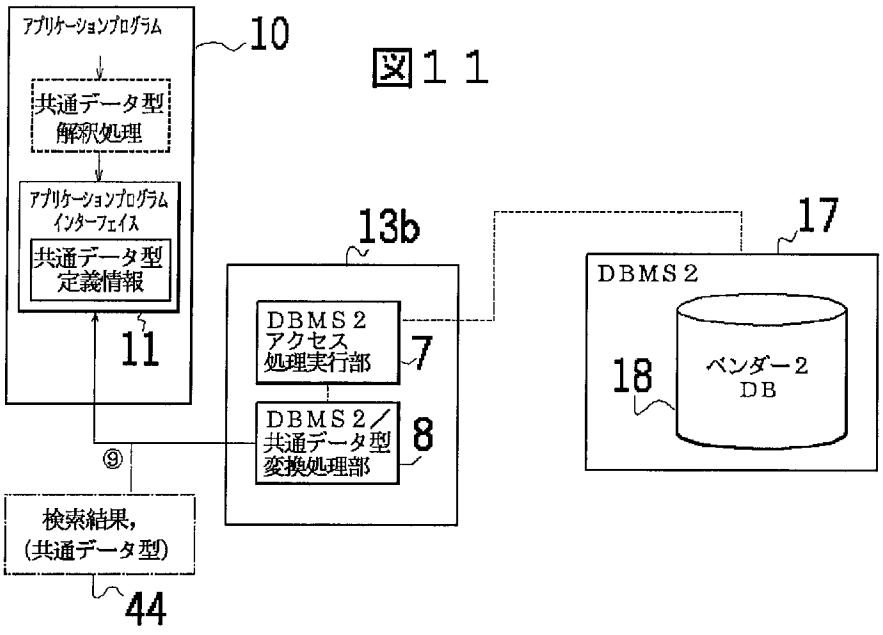


図１１

[Drawing 12]

図１２

51 RCAデータ型	52 ORACLE	53 informix	54 Sybase
RCAChar	CHAR	CHAR	CHAR
RCALongchar	LONG	CHAR	TEXT
RCABin	ROW	CHAR	BINARY
RCALongbin	LONGRAW	CHAR	IMAGE
RCAInt	NUMBER	INTEGER	INT
RCALong	NUMBER	DECIMAL	MONEY
RCAFloat	NUMBER	FLOAT	FLOAT
RCANumber	NUMBER	CECIMAL	FLOAT
RCADate	DATE	DATE	DATETIME

[Drawing 13]

図１３

ＯＲＡＣＬＥ→ＲＣＡデータ型

ＯＲＡＣＬＥ	ＲＣＡデータ型
ＣＨＡＲ	ＲＣＡＣｈａｒ
ＶＡＲＣＨＡＲ	ＲＣＡＣｈａｒ
ＬＯＮＧ	ＲＣＡＬｏｎｇｃｈａｒ
ＲＡＷ	ＲＣＡＣｈａｒ
ＬＯＮＧＲＡＷ	ＲＣＡＬｏｎｇｂｉｎ
ＮＵＭＢＥＲ	ＲＣＡＮｕｍｂｅｒ
ＤＡＴＥ	ＲＣＡｄａｔｅ
ＲＯＷＩＤ	ＲＣＡＲｏｗｉｄ

[Drawing 14]

図１４

ｉｎｆｏｒｍｉｘ→ＲＣＡデータ型

ｉｎｆｏｒｍｉｘ	ＲＣＡデータ型
ＣＨＡＲ	ＲＣＡＣｈａｒ
ＩＮＴＥＧＥＲ	ＲＣＡＩｎｔ
ＳＭＡＬＬＩＮＴ	ＲＣＡＩｎｔ
ＦＬＯＡＴ	ＲＣＡＦｌｏａｔ
ＳＭＡＬＬＦＬＯＡＴ	ＲＣＡＦｌｏａｔ
ＤＥＣＩＭＡＬ	ＲＣＡＮｕｍｂｅｒ
ＢＣＤ	ＲＣＡＮｕｍｂｅｒ
ＭＯＮＥＹ	ＲＣＡＮｕｍｂｅｒ
ＤＡＴＥ	ＲＣＡＤａｔｅ
ＳＥＲＩＡＬ	ＲＣＡＲｏｗｉｄ

[Drawing 15]

図１５

Ｓｙｂａｓｅ→RCAデータ型

Ｓｙｂａｓｅ	RCAデータ型
CHAR	RCAChar
VARCHAR	RCAChar
TEXT	RCALongchar
BINARY	RCABin
VARBINARY	RCABin
IMAGE	RCALongbin
INT	RCAInt
SMALLINT	RCAInt
TINYINT	RCAInt
BIT	RCAInt
MONEY	RCALong
FLOAT	RCAFloat
REAL	RCAFloat
DATETIME	RCADate
SMALLDATETIME	RCADate